

USER GUIDE

User Guide

Bold Reports

Version - v2.2.32 | Release Date - Jul 6,2020

| | |
|----------------------------------------------|-----|
| Overview | 116 |
| Key features | 116 |
| Create a support incident | 116 |
| Overview | 116 |
| Key features | 116 |
| Create a support incident | 117 |
| Overview | 117 |
| Registration..... | 117 |
| Manage Users and Groups..... | 118 |
| Manage User | 118 |
| Import Users | 118 |
| Manage Group | 118 |
| Import Groups..... | 118 |
| Manage Permission..... | 118 |
| Manage Report | 118 |
| Create Report..... | 118 |
| Edit Report | 119 |
| Upload Report..... | 119 |
| Update Report | 119 |
| Move,Clone Report..... | 119 |
| Copy Report | 119 |
| Share Report | 119 |
| Delete Report..... | 119 |
| Public Report..... | 119 |
| Manage Categories | 119 |
| Designing Reports | 119 |
| Connecting your data..... | 119 |
| Transform your data | 120 |
| Working with report parameters..... | 120 |
| Embed images into the report..... | 120 |
| Interacting with report design surface | 120 |
| Report items | 120 |
| Customize appearance..... | 120 |
| Report design settings | 120 |
| Shape report data | 121 |

| | |
|------------------------------------------------------------------------------------------------------|-----|
| Interactive features..... | 121 |
| Working with Expressions..... | 121 |
| Open report | 121 |
| Save report..... | 121 |
| Preview report | 121 |
| Manage Schedule..... | 121 |
| Synchronization Schedule..... | 121 |
| Notifications..... | 121 |
| Cloud Settings | 121 |
| Report Settings..... | 122 |
| Subscription | 122 |
| REST API | 122 |
| Frequently asked questions | 122 |
| List of IP addresses that need to be white listed for Bold Reports Cloud remote database access..... | 122 |
| Does Report Server store reports execution data to the server database..... | 122 |
| Overview | 123 |
| Key features | 123 |
| Create a support incident | 123 |
| Overview | 124 |
| Key features | 124 |
| Create a support incident | 124 |
| Overview | 124 |
| Prerequisites | 124 |
| Hardware Requirements..... | 124 |
| Software Requirements | 125 |
| Registration & Download..... | 125 |
| Installation and Deployment | 125 |
| Installation | 125 |
| IIS Express | 126 |
| IIS..... | 126 |
| How to change the binding in the Bold Reports On-premise | 126 |
| See Also..... | 127 |
| Installation and deployment..... | 127 |
| Installation | 127 |
| IIS Express | 128 |

| | |
|--------------------------------------------------------------------------|-----|
| IIS..... | 128 |
| How to change the binding in the Bold Reports On-premise | 128 |
| See Also..... | 129 |
| Installation and Deployment | 129 |
| Installation | 130 |
| IIS Express | 130 |
| IIS..... | 130 |
| How to change the binding in the Bold Reports On-premise | 131 |
| See Also..... | 132 |
| Upgrade Bold Reports On-Premise version from 1.x to latest | 132 |
| Steps to upgrade Bold Reports On-Premise | 132 |
| Upgrade Bold Reports On-Premise version from 1.x to latest | 132 |
| Steps to upgrade Bold Reports On-Premise | 132 |
| Migrate Bold Reports On-Premise | 133 |
| Prerequisites | 133 |
| Steps to run the data migration utility..... | 133 |
| Upgrade Bold Reports On-Premise version from 2.x to latest | 134 |
| Upgrading guidelines | 134 |
| Backup the existing data..... | 135 |
| Proceeding with upgrade | 135 |
| Steps to upgrade Bold Reports On-Premise | 135 |
| Azure | 136 |
| Create a Bold Reports Azure App Service using the ARM template..... | 136 |
| Configure a new Bold Reports | 138 |
| Upgrade Bold Reports Azure App Service version from v1.x to latest | 139 |
| Create a Bold Reports Azure App Service with latest version | 139 |
| File storage resources for migration..... | 139 |
| Azure Blob Storage resources for migration..... | 140 |
| Data migration with migration utility | 140 |
| Migrate Syncfusion Report Platform Report Server Azure App Service..... | 141 |
| Create a Bold Reports Azure App Service | 141 |
| File storage resources for migration..... | 141 |
| Azure Blob Storage resources for migration..... | 142 |
| Data migration with migration utility | 143 |
| Application Startup | 144 |

| | |
|------------------------------------------------------------------|-----|
| Database Configuration for Bold Reports Sites | 144 |
| Storage Type | 144 |
| Blob Storage..... | 145 |
| New User - System Administrator | 145 |
| Database Configuration for Bold Reports Server..... | 145 |
| Storage Type | 146 |
| Azure Active Directory Settings | 146 |
| Configure Active Directory settings | 147 |
| Azure Active Directory Settings | 147 |
| Email Settings..... | 148 |
| Database settings..... | 148 |
| Configure database connection details | 148 |
| SQL Server database | 149 |
| MySQL database | 149 |
| Oracle database | 149 |
| PostgreSQL database | 149 |
| Map database columns..... | 150 |
| Mark reports as public switch Allow/Restrict..... | 150 |
| Take Control Over the Public Reports..... | 151 |
| Mark reports as public switch Allow/Restrict Public Reports..... | 151 |
| Make Public..... | 151 |
| OAuth 2.0 Support in Boldreports | 152 |
| Prerequisites | 152 |
| Steps to configure OAuth 2.0 in BoldReports | 152 |
| OpenID Connect support in BoldReports..... | 153 |
| Prerequisites | 153 |
| Steps to configure OpenID Connect in BoldReports | 153 |
| Active Directory Synchronization Schedule | 153 |
| Email Notifications | 154 |
| Enable/Disable Synchronization schedule | 154 |
| Active Directory Synchronization Schedule | 154 |
| Email Notifications | 154 |
| Enable/Disable Synchronization schedule | 155 |
| Azure Active Directory Synchronization Schedule..... | 155 |
| Email Notifications | 155 |

| | |
|----------------------------------------------------------------|-----|
| Enable/Disable Synchronization schedule | 155 |
| Imported Existing Database Users Schedule Synchronization..... | 155 |
| Email Notifications | 156 |
| Enable/Disable Synchronization schedule | 156 |
| Create Report..... | 156 |
| Steps to create a report | 156 |
| Create Data | 156 |
| Add a chart report item | 157 |
| Assign Data..... | 157 |
| Customize the appearance | 159 |
| Save report..... | 159 |
| Preview report | 159 |
| See also | 159 |
| REST API Reference..... | 160 |
| Create Report..... | 160 |
| Steps to create a report | 160 |
| Create Data | 161 |
| Add a chart report item | 161 |
| Assign Data..... | 161 |
| Customize the appearance | 163 |
| Publish report..... | 163 |
| Preview report | 163 |
| See also | 164 |
| REST API Reference..... | 164 |
| Create Report..... | 164 |
| Steps to create a report | 164 |
| Create Data | 165 |
| Add a chart report item | 165 |
| Assign Data..... | 166 |
| Customize the appearance | 167 |
| Publish report..... | 167 |
| Preview report | 168 |
| See also | 168 |
| REST API Reference..... | 168 |
| Edit report in Report Designer..... | 168 |

| | |
|----------------------------------------------------|-----|
| Upload report or add report to Report Server | 169 |
| Steps to upload a report | 169 |
| Update report in the Report Server..... | 169 |
| Steps to update a Report | 169 |
| REST API Reference..... | 169 |
| Mark favorite report in Report server | 170 |
| Mark a Report as favorite | 170 |
| Remove a Report from favorites..... | 170 |
| Favorite Reports Category | 170 |
| Move, copy and clone reports in Report Server..... | 170 |
| Move Report | 170 |
| Copy Report | 171 |
| Clone Reports..... | 171 |
| Share report with users in the Report Server | 171 |
| Steps to share a Report..... | 171 |
| Remove Permission..... | 172 |
| share report link to the users..... | 172 |
| Get Link | 172 |
| Public report | 172 |
| Make public..... | 173 |
| Make Private | 173 |
| Public Reports | 173 |
| Delete report from Report Server..... | 173 |
| Delete All Sample Reports | 174 |
| REST API Reference..... | 174 |
| Version history of report..... | 174 |
| Versions..... | 174 |
| File logs | 174 |
| Download Report from Report Server..... | 175 |
| Manage the Report Views..... | 175 |
| Add the Report Views | 175 |
| Open the Report Views..... | 175 |
| Share the Report Views | 176 |
| Copy the Report Views link..... | 176 |
| Delete the Report Views | 176 |

| | |
|---------------------------------------------------------------|-----|
| Data sources management in Report Server | 176 |
| Add or create data sources | 176 |
| Steps to add a data source..... | 176 |
| Share data sources..... | 177 |
| Steps to share a data source | 177 |
| Update data sources | 178 |
| Steps to update the data source..... | 178 |
| Remove data source permission..... | 178 |
| Delete data source | 178 |
| REST API Reference..... | 179 |
| Dataset management in Report Server | 180 |
| Create dataset..... | 180 |
| Steps to create a dataset | 180 |
| Add or upload dataset | 180 |
| Create report with dataset | 181 |
| Edit dataset | 181 |
| Share dataset | 181 |
| Steps to share a dataset..... | 181 |
| Remove permission..... | 182 |
| Version history | 182 |
| Versions..... | 182 |
| File logs | 182 |
| Update dataset | 182 |
| Delete dataset..... | 183 |
| REST API Reference..... | 183 |
| Manage report schedules | 184 |
| Add schedule..... | 184 |
| Add schedule from `+` button menu | 184 |
| Add schedule from context menu of the respective reports..... | 184 |
| Add schedule from schedule listing page | 185 |
| Mail template customization | 186 |
| Mail template predefined variables..... | 186 |
| Edit schedule..... | 187 |
| Run now | 187 |
| Enable or disable schedule | 187 |

| | |
|---------------------------------------------------------------|-----|
| Delete schedules..... | 187 |
| REST API reference..... | 188 |
| Manage report schedules | 189 |
| Add schedule..... | 189 |
| Add schedule from `+` button menu | 189 |
| Add schedule from context menu of the respective reports..... | 189 |
| Add schedule from schedule listing page | 190 |
| Mail template customization..... | 191 |
| Mail template predefined variables..... | 192 |
| Edit schedule..... | 192 |
| Run now | 192 |
| Enable or disable schedule | 192 |
| Delete schedules..... | 193 |
| REST API reference..... | 193 |
| Schedule report settings..... | 194 |
| Enable file compression | 194 |
| Custom schedule report settings..... | 195 |
| Mail template customization..... | 195 |
| Mail template predefined variables..... | 195 |
| Manage users..... | 195 |
| Add new users..... | 196 |
| Edit users..... | 196 |
| Delete users | 196 |
| From user management page | 196 |
| From user edit page | 197 |
| Activate users..... | 197 |
| Deactivate users..... | 197 |
| Assign users to group..... | 197 |
| Manage permissions | 198 |
| Manage Users | 198 |
| Add new users..... | 198 |
| Add individual users..... | 198 |
| Edit users..... | 198 |
| Delete users | 198 |
| From user management page | 198 |

| | |
|------------------------------------------------|-----|
| From user edit page | 199 |
| Deactivate users..... | 199 |
| Activate users..... | 199 |
| Manage permissions | 199 |
| Assign users to group..... | 199 |
| Manage Users | 199 |
| Add new users..... | 199 |
| Add individual users..... | 200 |
| Edit users..... | 200 |
| Delete users | 200 |
| From user management page | 200 |
| From user edit page | 200 |
| Deactivate users..... | 200 |
| Activate users..... | 200 |
| Manage permissions | 201 |
| Assign users to group..... | 201 |
| Import users from CSV | 201 |
| Download users CSV template file..... | 201 |
| CSV template requirements..... | 202 |
| Add or import users from CSV template..... | 202 |
| Import users from CSV | 202 |
| Add users from CSV file..... | 203 |
| CSV template requirements..... | 203 |
| Import Active Directory users | 204 |
| Search users | 204 |
| Import users..... | 204 |
| Duplicate users..... | 204 |
| Modify Active Directory connection | 204 |
| Azure Active Directory User Import..... | 205 |
| Search Users..... | 205 |
| Import Users | 205 |
| Duplicate Users | 205 |
| Modify Azure Active Directory Connection | 205 |
| User Import from a Database | 206 |
| Listing Database Users | 206 |

| | |
|--------------------------------------------------------------------|-----|
| Select Users and Import..... | 206 |
| Modify Existing Database Connection | 206 |
| Synchronize Active Directory users | 206 |
| Synchronize users | 206 |
| Synchronization procedure..... | 207 |
| Duplicate users..... | 207 |
| Synchronize Active Directory users | 207 |
| Synchronize users | 207 |
| Synchronization procedure..... | 208 |
| Duplicate users..... | 208 |
| Azure Active Directory User Synchronization | 208 |
| Synchronize Users..... | 208 |
| Synchronization procedure..... | 209 |
| Duplicate Users | 209 |
| Synchronization of Imported Users From the Existing Database | 209 |
| Synchronize Users..... | 209 |
| Synchronization procedure..... | 209 |
| Duplicate Users | 210 |
| Manage Groups..... | 210 |
| Add new group..... | 210 |
| Edit group..... | 210 |
| Delete group | 210 |
| From group management page | 210 |
| From group edit page..... | 210 |
| Assign users..... | 210 |
| Manage permissions | 211 |
| Manage Groups..... | 211 |
| Add new group..... | 211 |
| Edit group..... | 211 |
| Delete group | 211 |
| From group management page | 211 |
| From group edit page..... | 211 |
| Assign users..... | 211 |
| Manage permissions | 212 |
| Import Active Directory Groups..... | 212 |

| | |
|----------------------------------------------------|-----|
| Search groups..... | 212 |
| Import groups | 212 |
| Duplicate groups..... | 212 |
| Import Active Directory Groups..... | 213 |
| Search groups..... | 213 |
| Import groups | 213 |
| Duplicate groups..... | 213 |
| Azure Active Directory Group Import | 213 |
| Search Groups..... | 214 |
| Import Groups..... | 214 |
| Duplicate Groups | 214 |
| Synchronize Active Directory Group..... | 214 |
| Synchronize groups..... | 215 |
| Synchronization procedure..... | 215 |
| Duplicate groups..... | 215 |
| Synchronize Active Directory Group..... | 215 |
| Synchronize groups..... | 216 |
| Synchronization procedure..... | 216 |
| Duplicate groups..... | 216 |
| Azure Active Directory Group Synchronization | 216 |
| Synchronize Groups | 217 |
| Synchronization procedure..... | 217 |
| Duplicate Groups | 217 |
| Account Activation..... | 217 |
| Automatic activation..... | 217 |
| Email activation..... | 218 |
| Manage Permissions..... | 218 |
| Access modes..... | 218 |
| Entity | 218 |
| Scope..... | 219 |
| Manage users permissions..... | 219 |
| Steps to add permission to users..... | 220 |
| Manage group permissions..... | 220 |
| Steps to add permission to the group..... | 220 |
| Manage Categories | 220 |

| | |
|------------------------------------------------------------------------|-----|
| Open category..... | 221 |
| Create category..... | 221 |
| Update category | 221 |
| Share category | 221 |
| Steps to share a category..... | 221 |
| Remove permission..... | 222 |
| Delete category..... | 222 |
| REST API Reference..... | 222 |
| Collaboration..... | 223 |
| Post a new comment | 223 |
| Reply to a comment | 223 |
| Edit a comment..... | 224 |
| Delete a comment..... | 224 |
| Show parent comment of a reply | 224 |
| Mention users in the comment | 224 |
| Notifications..... | 225 |
| Admin notification settings..... | 225 |
| System notifications..... | 225 |
| Mail notifications | 225 |
| Auto watch comments of created items | 225 |
| Auto watch comments of accessible items..... | 225 |
| Report schedule notification..... | 225 |
| User schedule notification | 225 |
| Default settings..... | 225 |
| Allow/restrict settings..... | 225 |
| User notification settings..... | 225 |
| Specific watch | 226 |
| Authentication settings..... | 226 |
| Single Sign-On | 226 |
| Single Sign-On(SSO) with OAuth 2.0 authentication in Bold Reports..... | 226 |
| Prerequisites | 226 |
| Steps to configure OAuth 2.0 in Bold Reports | 226 |
| Single Sign-On(SSO) with OneLogin authentication in Bold Reports..... | 227 |
| How to register the Bold Reports application in OneLogin | 228 |
| Prerequisites | 228 |

| | |
|-----------------------------------------------------------------------------|-----|
| Enable OneLogin authentication in Bold Reports..... | 229 |
| Single Sign-On(SSO) with Okta authentication in Bold Reports..... | 230 |
| How to register the Bold Reports application in Okta | 230 |
| Prerequisites | 230 |
| Steps to register the Bold Reports application | 230 |
| Enable Okta authentication in Bold Reports..... | 231 |
| Single Sign-On(SSO) with Amazon Cognito authentication in Bold Reports..... | 231 |
| How to register the Bold Reports application in Amazon Cognito | 232 |
| Prerequisites | 232 |
| Steps to register the Bold Reports application | 232 |
| Enable Amazon Cognito authentication in Bold Reports..... | 233 |
| Single Sign-On(SSO) with Auth0 authentication in Bold Reports | 234 |
| How to register the Bold Reports application in Auth0..... | 234 |
| Prerequisites | 234 |
| Steps to register the Bold Reports application | 234 |
| Enable Auth0 authentication in Bold Reports | 235 |
| Single Sign-On(SSO) with OpenID Connect authentication | 235 |
| Prerequisites | 235 |
| Steps to configure OpenID Connect in Bold Reports..... | 236 |
| Single Sign-On(SSO) with OneLogin authentication | 237 |
| How to register the Bold Reports application in OneLogin | 237 |
| Prerequisites | 237 |
| Steps to register the Bold Reports application | 237 |
| Enable OneLogin authentication in Bold Reports..... | 238 |
| Single Sign-On(SSO) with Okta authentication in Bold Reports..... | 238 |
| How to register the Bold Reports application in Okta | 238 |
| Prerequisites | 238 |
| Steps to register the Bold Reports application | 239 |
| Enable Okta authentication in Bold Reports..... | 239 |
| Single Sign-On(SSO) with Auth0 authentication in Bold Reports | 240 |
| How to register the Bold Reports application in Auth0..... | 240 |
| Prerequisites | 240 |
| Steps to register the Bold Reports application | 240 |
| Enable Auth0 authentication in BoldReports | 241 |
| Office 365 for authentication | 242 |

| | |
|-------------------------------------------------------------------------|-----|
| Embed in Application | 242 |
| REST API | 242 |
| Reporting Components | 242 |
| Embed URL | 242 |
| Embed Code of Report | 243 |
| See also | 243 |
| Get an embed code of report | 243 |
| Getting an embed URL | 243 |
| Steps to embed report using the embed URL | 243 |
| Public embedding | 244 |
| Private embedding | 244 |
| Report Server report using the Bold Reporting Tools Report Viewer | 244 |
| Integrate a Bold Reporting Tools Report Designer | 245 |
| Tenant Management | 245 |
| Creating a New Site | 245 |
| Site Creation | 246 |
| Select Database | 246 |
| Select Storage | 247 |
| Blob Storage | 247 |
| Select Administrator | 247 |
| Localization | 247 |
| Adding new localizations | 247 |
| Adding localizations to the Bold Report On premise | 247 |
| User Profile | 248 |
| View profile | 248 |
| Edit profile | 248 |
| Change password | 248 |
| My permissions | 248 |
| Edit notification settings | 249 |
| Custom rebranding | 249 |
| Organization name | 249 |
| Site URL | 249 |
| Login screen | 249 |
| Header | 249 |
| Favicon | 249 |

| | |
|---------------------------------------------|-----|
| Email..... | 249 |
| Display..... | 250 |
| Time zone..... | 250 |
| Date format..... | 250 |
| Time format | 250 |
| Powered by Syncfusion | 250 |
| Copyright information..... | 250 |
| Report Designer | 250 |
| Key features | 250 |
| Report Creation..... | 251 |
| Create Data | 251 |
| Edit dataset name | 252 |
| Drag and drop table in query designer | 252 |
| Execute query | 252 |
| Save Data | 252 |
| Add table report item | 252 |
| Assign data | 253 |
| Add column header..... | 253 |
| Assign data fields in table cell | 253 |
| Resize the column..... | 254 |
| Resize the row..... | 254 |
| Customize the appearance | 254 |
| Add total | 255 |
| Save report..... | 255 |
| Preview report | 255 |
| See also | 256 |
| Manage data | 256 |
| See also | 256 |
| Data source | 256 |
| Embedded data source | 257 |
| Shared data source | 257 |
| Custom data processing extensions | 257 |
| See also | 257 |
| Create an embedded data source..... | 257 |
| Add a new data source | 257 |

| | |
|--------------------------------------|-----|
| Modify an embedded data source..... | 258 |
| Link a shared datasource | 259 |
| Modify shared datasource | 260 |
| Delete a data source | 260 |
| Duplicate a data source | 260 |
| Report data sets..... | 261 |
| Embedded data set..... | 261 |
| Shared data set | 261 |
| See also | 261 |
| Create an embedded dataset | 261 |
| Design query data | 261 |
| Save Data | 263 |
| Modify an embedded dataset..... | 263 |
| Link a shared dataset | 263 |
| Modify shared dataset..... | 264 |
| Delete a dataset | 265 |
| Duplicate a dataset | 265 |
| Fields | 266 |
| Create calculated field | 266 |
| Set expression | 267 |
| Reset Expression | 267 |
| Create query field | 267 |
| Delete field..... | 268 |
| Change field name | 268 |
| Add a filter to a dataset | 269 |
| Create Data | 269 |
| Apply filters at dataset level | 269 |
| Add filters..... | 269 |
| Set expression | 269 |
| Remove filter..... | 269 |
| Example..... | 269 |
| Parameters..... | 270 |
| Open dataset parameters dialog | 270 |
| Create parameter..... | 270 |
| Edit parameter | 271 |

| | |
|----------------------------------------|-----|
| Delete parameter..... | 271 |
| Reorder parameters..... | 271 |
| Transforming data..... | 272 |
| Join Tables..... | 272 |
| Open query joiner dialog | 272 |
| Create a table relation | 272 |
| Table drop-down list | 273 |
| Join Types..... | 273 |
| Table fields | 274 |
| Operator..... | 274 |
| Create multiple join condition | 274 |
| Save a table relation | 275 |
| Edit a table relation..... | 275 |
| Delete a join condition..... | 275 |
| Delete a table relation | 276 |
| Create multiple table relation..... | 276 |
| Supported Join Types..... | 276 |
| Inner join..... | 276 |
| Left outer join | 277 |
| Right outer join | 278 |
| Full outer join..... | 279 |
| Formatting Columns | 279 |
| Rename a column | 279 |
| Exclude columns | 280 |
| Query Filters..... | 281 |
| Add filters..... | 281 |
| Remove Filters | 282 |
| Create or link dataset parameters..... | 282 |
| Open parameters dialog | 282 |
| Create a new dataset parameter | 283 |
| Set Expression..... | 283 |
| Reset Expression | 284 |
| Reorder parameters..... | 284 |
| Remove parameter | 285 |
| Link parameters manually..... | 285 |

| | |
|-----------------------------------------------------|-----|
| Create query and report parameters..... | 285 |
| Steps to link parameters | 285 |
| Define query parameters..... | 285 |
| Create query parameter in design mode..... | 286 |
| Create query parameter in code mode | 286 |
| Query expression | 287 |
| Open query expression dialog | 287 |
| Create an expression column..... | 287 |
| Edit an expression column..... | 288 |
| Delete an expression column..... | 288 |
| Close query expression dialog | 288 |
| Supported built-in functions | 288 |
| Numbers..... | 288 |
| Conditional..... | 289 |
| Logical | 290 |
| Date..... | 290 |
| String | 291 |
| Report Parameters..... | 292 |
| Data set query parameters | 292 |
| Cascading parameter | 292 |
| See also | 292 |
| Add report parameter to the report..... | 292 |
| Create parameter..... | 292 |
| Filter a table data based on report parameter | 294 |
| Edit a report parameter | 294 |
| Delete a report parameter..... | 295 |
| Define available values for a parameter..... | 295 |
| Manual values | 296 |
| Query values | 296 |
| Remove available values..... | 297 |
| Filter a table data based on report parameter | 297 |
| Define default values for a parameter..... | 298 |
| Manual values | 298 |
| Query values | 299 |
| Remove default values..... | 299 |

| | |
|-----------------------------------------------------|-----|
| Create a multi value report parameter..... | 300 |
| Filter a table data based on report parameter | 300 |
| Cascading Parameter | 301 |
| Steps to create cascading parameters..... | 301 |
| Create the main dataset | 301 |
| Create dataset for independent parameter | 303 |
| Set available values for independent parameter..... | 303 |
| Create dataset for dependent parameter | 304 |
| Set available values for dependent parameter..... | 305 |
| Filter a table data based on report parameter | 305 |
| Parameters Layout..... | 305 |
| Order parameters | 306 |
| Insert row or column | 306 |
| Delete row or column | 307 |
| Image manager | 307 |
| See also | 307 |
| Image Manager | 308 |
| Add an embedded image | 308 |
| Design a report using an embedded image | 308 |
| Delete an embedded image..... | 309 |
| Compose report | 309 |
| Toolbar | 309 |
| Properties panel..... | 309 |
| Multi select report item properties | 309 |
| Filtering | 309 |
| Sorting | 309 |
| Grouping | 309 |
| Drill through reports | 310 |
| Hyperlink..... | 310 |
| Drilldown action..... | 310 |
| Expression builder..... | 310 |
| Format data..... | 310 |
| Unit switcher..... | 310 |
| Code module..... | 310 |
| Toolbar | 310 |

| | |
|-----------------------------------------------|-----|
| New | 310 |
| Open..... | 311 |
| Save | 311 |
| Cut | 311 |
| Copy | 311 |
| Paste..... | 311 |
| Delete..... | 311 |
| Undo..... | 311 |
| Redo | 311 |
| Zoom out..... | 311 |
| Zoom in | 311 |
| Layout ordering..... | 311 |
| Alignment..... | 311 |
| Distribute | 311 |
| Sizing | 311 |
| Align to grid..... | 312 |
| Size to grid..... | 312 |
| View | 312 |
| Preview | 312 |
| Design surface..... | 312 |
| Key features | 312 |
| See also | 313 |
| Report item resizing..... | 313 |
| Resize using resizer | 313 |
| Resize using width and height properties..... | 314 |
| Resize using keyboard shortcuts..... | 314 |
| Using touch resizer..... | 314 |
| Report item selection..... | 315 |
| Using mouse action..... | 315 |
| Using keyboard shortcuts | 315 |
| Report item alignment..... | 315 |
| Align | 315 |
| Left align..... | 316 |
| Center align..... | 316 |
| Right align | 316 |

| | |
|------------------------------------------|-----|
| Top align..... | 316 |
| Middle align | 317 |
| Bottom align..... | 317 |
| Center horizontally..... | 317 |
| Center vertically..... | 317 |
| Distribute | 318 |
| Horizontally..... | 318 |
| Vertically | 318 |
| Sizing | 318 |
| Same size..... | 318 |
| Same height | 318 |
| Same width | 318 |
| Align to grid..... | 318 |
| Size to grid..... | 318 |
| Context menu | 319 |
| Header and footer..... | 319 |
| Add Header | 319 |
| Add Footer | 319 |
| Remove Header..... | 319 |
| Remove Footer..... | 319 |
| Insert | 319 |
| Insert item at design surface | 319 |
| Insert item at rectangle container | 320 |
| Insert item at tablix cell | 320 |
| Cut | 320 |
| Cut item from design surface..... | 320 |
| Cut item from rectangle container | 320 |
| Cut item from tablix cell..... | 320 |
| Copy | 321 |
| Copy item from design surface | 321 |
| Copy item from rectangle container..... | 321 |
| Copy item from tablix cell..... | 321 |
| Paste..... | 322 |
| Paste item at design area..... | 322 |
| Paste item at rectangle | 322 |

| | |
|------------------------------------|-----|
| Paste item at tablix cell | 322 |
| Delete..... | 322 |
| Properties Panel..... | 322 |
| Open properties panel | 322 |
| Header properties..... | 323 |
| Body properties..... | 323 |
| Footer properties | 323 |
| Report item properties | 323 |
| Common properties | 323 |
| Set expression | 323 |
| Reset expression | 324 |
| Advanced properties..... | 324 |
| Dependent properties..... | 324 |
| Report properties..... | 325 |
| Basic settings..... | 325 |
| Border | 325 |
| Background color | 325 |
| Background Image | 325 |
| Code | 325 |
| Page Units | 325 |
| Margin..... | 325 |
| Paper Size..... | 326 |
| Orientation..... | 326 |
| Paper size | 326 |
| Language | 326 |
| Miscellaneous | 326 |
| Report Sections | 326 |
| Report Header and Footer | 326 |
| Report Body | 327 |
| General properties..... | 327 |
| Border | 327 |
| Background color | 327 |
| Background Image | 327 |
| Header and Footer properties | 327 |
| Height..... | 327 |

| | |
|---------------------------------------------------|-----|
| Print on first page..... | 327 |
| Print on last page | 327 |
| Body properties..... | 327 |
| Size | 327 |
| Report Header and Footer | 328 |
| Show or hide report header..... | 328 |
| Show or hide report footer..... | 328 |
| Common properties..... | 328 |
| Name..... | 328 |
| Border properties..... | 329 |
| Border style..... | 329 |
| Border color | 329 |
| Border width | 329 |
| Setting borders for each side | 329 |
| Set border properties based on dynamic value..... | 330 |
| Background color | 330 |
| Visibility..... | 330 |
| Position | 330 |
| Custom properties | 330 |
| Tooltip | 331 |
| Report Item Multiselection..... | 331 |
| Open properties panel | 331 |
| Same item type selection..... | 332 |
| Different item type selection..... | 332 |
| Cell with same item type | 333 |
| Cell with different item type..... | 333 |
| Sorting | 334 |
| Add Sorting | 334 |
| Set Expression | 334 |
| Reset Expression | 335 |
| Reordering | 335 |
| Remove Sorting..... | 335 |
| Grouping | 335 |
| Add Grouping | 335 |
| Set Expression | 336 |

| | |
|------------------------------------------------------|------------|
| Reset Expression | 336 |
| Reordering | 337 |
| Remove Grouping | 337 |
| Filters..... | 337 |
| Add filters..... | 337 |
| Set Expression | 338 |
| Reset Expression | 338 |
| Reordering | 338 |
| Remove Filters | 339 |
| Format..... | 339 |
| Format Numbers..... | 339 |
| Format Currency | 339 |
| Format Date | 340 |
| Format Time..... | 340 |
| Format Scientific | 340 |
| Format Percentage | 340 |
| Format Custom | 340 |
| Linking | 341 |
| Report Linking | 341 |
| Report Path | 341 |
| Hyperlink..... | 343 |
| Add a hyperlink | 343 |
| Background image | 343 |
| Source | 343 |
| Value | 343 |
| MIME Type..... | 344 |
| Background repeat..... | 344 |
| Add company logo as the background to a report | 344 |
| Create SSRS drilldown report..... | 345 |
| Create data..... | 345 |
| Add drilldown action to a table group | 345 |
| Show group on initial display..... | 346 |
| Hide group on initial display | 346 |
| Create SSRS drill through report | 347 |
| Create data..... | 347 |

| | |
|------------------------------------------|-----|
| Design drill through report | 347 |
| Add drill through action to a chart..... | 348 |
| Drill through chart series | 349 |
| Create Multiple-Column Report | 349 |
| Properties..... | 349 |
| Preview | 350 |
| Interactive sorting..... | 350 |
| Create data..... | 350 |
| Add interactive sorting..... | 350 |
| Preview | 351 |
| Unit Switcher..... | 351 |
| Supported Unit Types | 351 |
| Layout Ordering | 352 |
| Send Backward..... | 352 |
| Bring Forward..... | 352 |
| Send To Back | 352 |
| Bring To Front | 353 |
| Expression Builder..... | 353 |
| Supported Expressions..... | 353 |
| Dataset Fields in Expressions | 353 |
| Displaying Fields Collection..... | 354 |
| Parameters in Expressions | 354 |
| Displaying Parameters Collection | 354 |
| Built-in Fields..... | 354 |
| Global Collection..... | 354 |
| User Collection | 355 |
| Built-in Functions | 355 |
| Text Functions..... | 355 |
| Date Time Functions | 362 |
| Math Functions | 368 |
| Inspection Functions | 371 |
| Program Flow Functions | 371 |
| Aggregate Functions | 372 |
| Financial Functions..... | 375 |
| Conversion Functions..... | 379 |

| | |
|--------------------------------------------------------------|-----|
| Miscellaneous Functions..... | 380 |
| Operators in Expressions | 380 |
| Arithmetic | 380 |
| Comparison..... | 380 |
| String Concatenation | 381 |
| Logical and Bitwise..... | 381 |
| Bit Shift..... | 381 |
| Code Module..... | 382 |
| Add Custom Code to a report | 382 |
| Add Assembly References..... | 383 |
| Create a custom assembly | 383 |
| Adding a references to custom assembly | 383 |
| Add Class instances..... | 384 |
| Page layout..... | 384 |
| Rendering layouts | 384 |
| Pagination | 385 |
| Paper size | 385 |
| Margin | 386 |
| Avoid the extra blank pages in print and print preview | 386 |
| Horizontal usable area | 386 |
| Vertical usable area..... | 386 |
| Report Variables..... | 387 |
| Add a report variable | 387 |
| Delete a report variable | 388 |
| Call report variable in a report..... | 388 |
| Configure report items..... | 388 |
| See also | 388 |
| Line..... | 389 |
| To draw a line..... | 389 |
| Line Properties | 389 |
| Line Styles | 389 |
| Border | 389 |
| Position | 390 |
| Size | 390 |
| Visibility..... | 390 |

| | |
|----------------------------------------|-----|
| Miscellaneous | 390 |
| Rectangle | 390 |
| Add a rectangle to the report | 390 |
| Properties..... | 390 |
| Basic Settings | 390 |
| Page break..... | 391 |
| Position | 391 |
| Visibility..... | 391 |
| Miscellaneous | 391 |
| Set expression | 392 |
| Reset expression | 392 |
| Advanced properties..... | 392 |
| Image | 392 |
| Add a image to the report..... | 392 |
| Image properties..... | 393 |
| Basic Settings | 393 |
| Link..... | 393 |
| Appearance | 394 |
| Size | 394 |
| Position | 394 |
| Visibility..... | 394 |
| Miscellaneous | 394 |
| Subreport | 395 |
| Properties..... | 395 |
| Basic Settings | 395 |
| Appearance | 395 |
| No Rows | 395 |
| Position | 397 |
| Visibility..... | 397 |
| Miscellaneous | 397 |
| Set expression..... | 398 |
| Reset expression | 398 |
| Advanced properties..... | 398 |
| Design rdl report using subreport..... | 398 |
| Design rdl report using subreport..... | 398 |

| | |
|----------------------------------------------------|-----|
| Create a main report..... | 398 |
| Create a subreport..... | 399 |
| Create dataset..... | 399 |
| Design report | 400 |
| Assign fields..... | 401 |
| Link sub report into main report..... | 401 |
| Set Parameters values..... | 402 |
| TextBox | 402 |
| Add a textbox to the report..... | 402 |
| Properties..... | 403 |
| Common properties | 403 |
| Textbox properties..... | 405 |
| Selected Text properties | 407 |
| Markup Type | 407 |
| Set expression..... | 407 |
| Reset expression | 407 |
| Advanced properties..... | 407 |
| Design RDL report using textbox..... | 407 |
| Design RDL report using textbox..... | 408 |
| Add textbox to the report..... | 408 |
| Display dynamic text using expression | 408 |
| Edit the expression..... | 408 |
| Associate a textbox with dataset..... | 409 |
| Drag and drop dataset field | 409 |
| Assign the field using expression editor | 409 |
| Associate a textbox with parameter | 409 |
| Drag and drop parameter field | 409 |
| Assign the parameter using expression editor | 410 |
| Configure and Format textbox content | 410 |
| Position and Sizing | 410 |
| Style textbox content..... | 410 |
| Format value in textbox | 411 |
| Report linking | 412 |
| Over all textbox..... | 412 |
| Selected text | 413 |

| | |
|------------------------------------|-----|
| Add hyperlink | 413 |
| Over all textbox..... | 413 |
| Selected text | 414 |
| Add HTML in to a report | 414 |
| Format plain text as HTML..... | 414 |
| Tablix | 415 |
| Table..... | 415 |
| Matrix..... | 416 |
| List | 416 |
| Tablix sections..... | 416 |
| Groups and total | 416 |
| Properties..... | 416 |
| Data | 416 |
| Appearance | 417 |
| Page break..... | 417 |
| Headers | 418 |
| Position | 418 |
| Visibility..... | 418 |
| Miscellaneous | 418 |
| Set expression | 418 |
| Reset expression | 418 |
| Advanced properties..... | 419 |
| Design RDL report using table..... | 419 |
| Grouping Panel | 419 |
| Show or hide grouping panel..... | 419 |
| Resize grouping panel..... | 419 |
| Modes of grouping panel..... | 419 |
| Default Mode | 419 |
| Advanced Mode | 419 |
| Visual Cues | 420 |
| Groups..... | 420 |
| Details Group | 420 |
| Label convention..... | 420 |
| Tablix member properties | 420 |
| Static member properties..... | 421 |

| | |
|-------------------------------------------------|-----|
| Group member properties..... | 421 |
| Add groups or total..... | 421 |
| Add parent group..... | 421 |
| Add child group..... | 422 |
| Add adjacent group..... | 422 |
| Add total | 423 |
| Delete Group..... | 423 |
| Tablix member properties | 423 |
| Static member properties | 423 |
| Miscellaneous | 423 |
| Group member properties..... | 425 |
| Basic settings..... | 425 |
| Groups..... | 425 |
| Miscellaneous | 425 |
| Page break..... | 426 |
| Assign data to tablix data region | 427 |
| Assign dataset | 427 |
| Assign data from data assign menu | 427 |
| Assign data from properties panel..... | 427 |
| Assign fields to tablix cell | 428 |
| Drag and drop data fields..... | 428 |
| Assign fields using data assign menu | 428 |
| Textbox properties..... | 429 |
| Assign or edit expression into table cell | 429 |
| Set expression using data assign menu | 429 |
| Set expression in properties panel | 430 |
| Edit expression using data assign menu | 430 |
| Edit expression in properties panel | 431 |
| Tablix cell..... | 431 |
| Change an item within a cell..... | 431 |
| Drag and drop report item into table cell..... | 431 |
| Insert item using cell menu..... | 432 |
| Cell properties..... | 432 |
| Delete an item from a cell..... | 432 |
| Cut Copy and Paste cell contents..... | 433 |

| | |
|----------------------------------------------------|-----|
| Cut Copy and Paste operation in single cell..... | 433 |
| Cut Copy and Paste operation in multiple cell..... | 434 |
| Tablix cell border..... | 435 |
| Border behaviour | 435 |
| Resize tablix data region | 436 |
| Over all data region..... | 436 |
| Resize using resizer | 436 |
| Set width and height in table properties | 437 |
| Resize the column..... | 438 |
| Set column width using gripper | 438 |
| Set column width in cell properties | 438 |
| Using touch resizer..... | 438 |
| Resize the row..... | 438 |
| Set row height using gripper | 438 |
| Set row height in cell properties..... | 439 |
| Using touch resizer..... | 439 |
| Insert or Delete a Row | 439 |
| Insert a row | 439 |
| Insert a row in a group..... | 440 |
| Delete a row..... | 441 |
| Delete a row from a group..... | 442 |
| Insert or Delete a Column..... | 442 |
| Insert a column | 442 |
| Insert a column in a group | 443 |
| Delete a column | 444 |
| Delete a column from a group | 444 |
| Tablix Group..... | 445 |
| Create groups..... | 445 |
| Group types..... | 445 |
| Edit group properties..... | 445 |
| Group scope..... | 446 |
| Insert or Delete a Row Group | 446 |
| Insert a row group..... | 446 |
| Parent row group | 446 |
| Child row group..... | 447 |

| | |
|-------------------------------------------------|-----|
| Adjacent row group | 448 |
| Delete row group | 449 |
| Edit group properties | 449 |
| Insert or Delete a Column Group..... | 449 |
| Insert a column group..... | 449 |
| Parent column group | 450 |
| Delete column group | 450 |
| Edit group properties | 451 |
| Add or Delete a Details Group | 451 |
| Add details group..... | 451 |
| Delete details group..... | 452 |
| Edit group properties | 452 |
| Merge Cells | 452 |
| Merge restrictions..... | 452 |
| Merge cells..... | 452 |
| Corner area | 453 |
| Body area | 453 |
| Row group..... | 453 |
| Column group..... | 454 |
| Split cells | 454 |
| Add filters to tablix data region | 454 |
| Set filter on tablix data region | 455 |
| Set filter on a tablix group | 455 |
| Sort data in a tablix data region..... | 456 |
| Set sort expression for tablix data region..... | 456 |
| Set sort expression on a tablix group..... | 456 |
| Add Total..... | 457 |
| Add total tablix body..... | 457 |
| Add total for row group | 457 |
| Add total for column group | 458 |
| Add Group Header and Footer | 458 |
| Format header and footer | 459 |
| Merge header cells..... | 460 |
| Merge footer cells..... | 460 |
| Edit footer cell content | 460 |

| | |
|-----------------------------------------------------|-----|
| Edit header cell content..... | 460 |
| Report preview | 460 |
| Nested Data Regions..... | 461 |
| Assign data to the nested data region..... | 461 |
| Assign data to the nested chart region..... | 461 |
| Assign data to the nested tablix region | 461 |
| Scope of data for nested data regions..... | 462 |
| Design a simple report with nested data region..... | 462 |
| Design ssrs rdl report using table..... | 464 |
| Add a table to the report | 464 |
| Initial design | 464 |
| Assign data fields | 464 |
| Drag and drop data fields..... | 464 |
| Assign fields using data assign menu..... | 465 |
| Textbox properties..... | 465 |
| Set header text..... | 465 |
| Using data assign menu | 465 |
| Set text in content property..... | 466 |
| Resize the column..... | 466 |
| Resize the row..... | 466 |
| Final design | 467 |
| Add Grouping and Totals | 467 |
| Create dataset..... | 467 |
| Simple table design..... | 467 |
| Add group data | 467 |
| Parent row group | 467 |
| Child row group..... | 468 |
| Add Totals | 469 |
| Add total yearly sales of a product | 469 |
| Add totals yearly sales of product category | 469 |
| Add the grand total to the report | 469 |
| Edit cell content | 470 |
| Format total row..... | 470 |
| Design and preview..... | 471 |
| Conditional Formatting..... | 471 |

| | |
|------------------------------------------|-----|
| Create dataset..... | 471 |
| Create parameter..... | 471 |
| New parameter creation..... | 471 |
| Assign value..... | 471 |
| Parameter list view | 472 |
| Add table data region | 472 |
| Formatting table | 472 |
| Apply conditional formatting..... | 472 |
| Preview report | 473 |
| Repeat Headers on Each Page in SSRS..... | 473 |
| Matrix..... | 474 |
| Properties..... | 474 |
| Add matrix to the report..... | 474 |
| Add row and column groups..... | 474 |
| Design SSRS Matrix Report | 475 |
| Create dataset..... | 475 |
| Add matrix to the report..... | 475 |
| Add parent row group..... | 476 |
| Add parent column group..... | 476 |
| Delete rows and columns | 477 |
| Delete details group..... | 477 |
| Add child row group..... | 478 |
| Add child column group..... | 478 |
| Calculate a summary..... | 479 |
| Format data..... | 479 |
| Change width or height..... | 479 |
| Merge matrix cells | 479 |
| Format matrix design..... | 480 |
| Report preview | 480 |
| List | 480 |
| Add report items in list | 480 |
| Properties..... | 481 |
| Data..... | 481 |
| Appearance | 481 |
| Position | 481 |

| | |
|----------------------------------------|-----|
| Visibility..... | 481 |
| Miscellaneous | 482 |
| Set expression | 482 |
| Reset expression | 482 |
| Advanced properties..... | 482 |
| Design RDL report using list | 482 |
| Design ssrs rdl report using list..... | 482 |
| Create dataset..... | 482 |
| Configure a list | 483 |
| Initial design | 483 |
| Add report items..... | 483 |
| Report header | 484 |
| Final design | 485 |
| Report preview | 485 |
| Chart..... | 485 |
| Add chart to the report..... | 485 |
| Chart parts | 485 |
| Column Chart | 486 |
| Add chart to the report..... | 486 |
| Create data..... | 486 |
| Assign data | 486 |
| Format column chart | 488 |
| General Settings..... | 488 |
| Basic Settings | 488 |
| Appearance | 491 |
| Chart Area | 491 |
| Title | 491 |
| Category Axis..... | 491 |
| Value Axis..... | 491 |
| Grid line..... | 491 |
| Page break..... | 492 |
| Miscellaneous | 492 |
| Preview report | 493 |
| Pie Chart..... | 493 |
| Add chart to the report..... | 493 |

| | |
|------------------------------|-----|
| Create data..... | 493 |
| Assign data | 494 |
| Format pie chart | 495 |
| General Settings..... | 496 |
| Basic Settings | 496 |
| Appearance | 498 |
| Chart Area | 498 |
| Title | 498 |
| Page break..... | 498 |
| Miscellaneous | 499 |
| Preview report | 499 |
| Bubble Chart | 499 |
| Add chart to the report..... | 499 |
| Create data..... | 500 |
| Assign data | 500 |
| Format bubble chart | 502 |
| General Settings..... | 502 |
| Basic Settings | 502 |
| Appearance | 504 |
| Chart Area | 504 |
| Title | 505 |
| Category Axis..... | 505 |
| Value Axis..... | 505 |
| Grid line..... | 505 |
| Page break..... | 506 |
| Miscellaneous | 506 |
| Preview report | 506 |
| Scatter Chart | 507 |
| Add chart to the report..... | 507 |
| Create data..... | 507 |
| Assign data | 507 |
| Format scatter chart | 509 |
| General Settings..... | 509 |
| Basic Settings | 510 |
| Appearance | 512 |

| | |
|------------------------------|-----|
| Chart Area | 512 |
| Title | 512 |
| Category Axis..... | 512 |
| Value Axis..... | 512 |
| Grid line..... | 513 |
| Page break..... | 513 |
| Miscellaneous | 514 |
| Preview report | 514 |
| Line Chart..... | 514 |
| Add chart to the report..... | 514 |
| Create data..... | 514 |
| Assign data | 515 |
| Format line chart..... | 516 |
| General Settings..... | 517 |
| Basic Settings | 517 |
| Appearance | 519 |
| Chart Area | 519 |
| Title | 519 |
| Category Axis..... | 520 |
| Value Axis..... | 520 |
| Grid line..... | 520 |
| Page break..... | 520 |
| Miscellaneous | 521 |
| Preview report | 521 |
| Area Chart..... | 521 |
| Add chart to the report..... | 521 |
| Create data..... | 521 |
| Assign data | 522 |
| Format area chart | 523 |
| General Settings..... | 524 |
| Basic Settings | 524 |
| Appearance | 526 |
| Chart Area | 526 |
| Title | 527 |
| Category Axis..... | 527 |

| | |
|------------------------------|-----|
| Value Axis..... | 527 |
| Grid line..... | 527 |
| Page break..... | 528 |
| Miscellaneous | 528 |
| Preview report | 528 |
| Radar Chart | 529 |
| Add chart to the report..... | 529 |
| Create data..... | 529 |
| Assign data | 529 |
| Format line chart..... | 531 |
| General Settings..... | 531 |
| Basic Settings | 531 |
| Appearance | 534 |
| Chart Area | 534 |
| Title | 534 |
| Category Axis..... | 534 |
| Value Axis..... | 534 |
| Grid line..... | 534 |
| Page break..... | 535 |
| Miscellaneous | 535 |
| Preview report | 535 |
| Chart Legend..... | 536 |
| Show or hide legend | 536 |
| Format legend..... | 536 |
| Show border..... | 536 |
| Background color | 537 |
| Font | 537 |
| Font Style and Weight..... | 537 |
| Title | 537 |
| Legend position..... | 538 |
| Enable custom bounds..... | 538 |
| Chart Marker..... | 538 |
| Show or hide marker..... | 538 |
| Format marker | 539 |
| Border | 539 |

| | |
|------------------------------------|-----|
| Color..... | 539 |
| Marker type | 539 |
| Size | 539 |
| Chart Data Label..... | 540 |
| Show or hide data label | 540 |
| Format data label..... | 540 |
| Show border..... | 540 |
| Background color | 541 |
| Font | 541 |
| Font Style and Weight..... | 541 |
| Position | 541 |
| Label Rotation | 542 |
| Format..... | 542 |
| Label..... | 543 |
| Use Value as Label | 543 |
| Chart Smart Label..... | 544 |
| Enable or disable smart label..... | 544 |
| Format smart label..... | 544 |
| Label Style | 544 |
| Value | 544 |
| Chart Axis | 545 |
| Show or hide axis | 545 |
| Axis Title | 546 |
| Line Style | 546 |
| Label Overflow Mode..... | 546 |
| Trim | 546 |
| Hide | 546 |
| Label Rotation | 546 |
| Label Format | 547 |
| Label Font..... | 547 |
| Tick | 547 |
| Major Ticks..... | 547 |
| Minor Ticks..... | 548 |
| Tick Position | 548 |
| Chart types..... | 548 |

| | |
|----------------------------------------------------|-----|
| Comparison..... | 549 |
| Column..... | 549 |
| Bar | 549 |
| Proportion..... | 549 |
| Pie..... | 549 |
| Shape..... | 549 |
| Distribution | 550 |
| Area..... | 550 |
| Line..... | 550 |
| Scatter..... | 550 |
| Polar | 550 |
| Chart Nested Data Regions..... | 551 |
| Create dataset..... | 551 |
| Add table to the report..... | 551 |
| Add parent row group..... | 552 |
| Add parent column group..... | 552 |
| Delete rows and columns | 553 |
| Delete details group..... | 553 |
| Add chart into the matrix..... | 554 |
| Assign data to chart | 554 |
| Format matrix | 555 |
| Format chart | 555 |
| Final design | 555 |
| Report preview | 555 |
| Add filters to chart data region..... | 556 |
| Set filter on a chart category group..... | 556 |
| Sort data in a chart data region | 556 |
| Set sort expression on a chart category group..... | 557 |
| Indicator..... | 557 |
| Add an indicator to the report..... | 557 |
| Properties..... | 558 |
| Basic settings..... | 558 |
| Position | 558 |
| Data | 558 |
| Indicator value | 558 |

| | |
|---------------------------------------|-----|
| Indicator types | 559 |
| Indicator states | 559 |
| Visibility..... | 559 |
| Miscellaneous | 560 |
| Set expression | 560 |
| Reset expression | 560 |
| Advanced properties..... | 560 |
| Gauge | 560 |
| Gauge types | 560 |
| See Also..... | 560 |
| Linear gauge..... | 560 |
| Add a linear gauge to the report..... | 560 |
| Properties..... | 561 |
| Basic settings..... | 561 |
| Data..... | 561 |
| Pointer..... | 562 |
| Scale | 562 |
| Tick mark..... | 563 |
| Range | 563 |
| Position | 564 |
| Visibility..... | 564 |
| Miscellaneous | 564 |
| Radial gauge..... | 564 |
| Add a radial gauge to the report..... | 564 |
| Properties..... | 565 |
| Basic settings..... | 565 |
| Data..... | 565 |
| Pointer..... | 566 |
| Scale | 567 |
| Tick mark..... | 567 |
| Range | 568 |
| Position | 568 |
| Visibility..... | 568 |
| Miscellaneous | 568 |
| Data bar..... | 568 |

| | |
|-----------------------------------------------|-----|
| Add data bar to the report..... | 569 |
| Create data..... | 569 |
| Assign data | 569 |
| Properties..... | 571 |
| General Settings..... | 571 |
| Basic Settings | 571 |
| Appearance | 573 |
| Chart Area | 573 |
| Page break..... | 573 |
| Position | 573 |
| Visibility..... | 573 |
| Miscellaneous | 574 |
| Set expression..... | 574 |
| Reset expression | 574 |
| Advanced properties..... | 574 |
| Design ssrs data bar report using table | 574 |
| Create dataset..... | 574 |
| Add data bar report item | 575 |
| Assign data | 576 |
| Configure properties..... | 576 |
| Report preview | 577 |
| Sparkline | 577 |
| Add sparkline to the report..... | 577 |
| Create data..... | 577 |
| Assign data | 578 |
| Properties..... | 579 |
| General Settings..... | 579 |
| Basic Settings | 579 |
| Appearance | 581 |
| Chart Area | 581 |
| Page break..... | 581 |
| Position | 582 |
| Visibility..... | 582 |
| Miscellaneous | 582 |
| Set expression..... | 582 |

| | |
|------------------------------------------------------|-----|
| Reset expression | 582 |
| Advanced properties..... | 582 |
| Design ssrs sparkline report using table | 582 |
| Create dataset..... | 582 |
| Add sparkline report item | 583 |
| Assign data | 584 |
| Configure properties..... | 585 |
| Report preview | 585 |
| Barcode | 585 |
| One-Dimensional barcodes..... | 585 |
| Adding a one dimensional barcode to the report..... | 586 |
| Properties..... | 586 |
| Two-Dimensional barcodes | 587 |
| Adding a two dimensional barcode to the report | 587 |
| Properties..... | 587 |
| General properties | 588 |
| Name..... | 588 |
| Appearance | 588 |
| Position | 588 |
| Visibility..... | 589 |
| Tooltip | 589 |
| Set expression | 589 |
| Reset expression | 589 |
| Advanced properties..... | 589 |
| Design ssrs rdl report using barcode | 589 |
| Create dataset..... | 589 |
| Add a table to the report | 589 |
| Configure table..... | 590 |
| Add barcode report item | 590 |
| Encode value in barcode | 590 |
| Customize appearance..... | 591 |
| Format table..... | 591 |
| Report header | 591 |
| Final design | 592 |
| Report preview | 592 |

| | |
|---------------------------------------------------------------------------------------------------------------------------------|-----|
| Supported barcode types..... | 592 |
| Open report | 593 |
| From Device | 593 |
| From Server..... | 593 |
| Save a Report | 593 |
| Save a report into report server | 594 |
| Save report to your computer | 594 |
| Preview a Report..... | 594 |
| Supported and Unsupported Items in Web Report Designer..... | 595 |
| Report Items | 595 |
| Features | 596 |
| DataSources | 596 |
| Keyboard Shortcuts..... | 596 |
| Shortcut keys..... | 596 |
| How to startup the Bold Reports On-Premise Edition using the newly added bindings from IIS | 597 |
| How to map a domain name for Bold Report On-Premise Edition before startup | 598 |
| How to resolve unavailable error with Bold Reports On-Premises Edition when it is occurred in reason of domain name change..... | 598 |
| How to setup the staging and production environment for Bold Reports On-Premise Edition..... | 599 |
| Prepare a staging environment | 599 |
| Staging to production | 600 |
| Copy resources..... | 600 |
| Create a Bold Reports Site | 600 |
| Add a reporting application | 600 |
| API application for Bold Reports IDP | 600 |
| UMS application for Bold Reports IDP | 601 |
| Windows Authentication application for Bold Reports IDP..... | 601 |
| API application for reporting application..... | 601 |
| Jobs application for reporting application | 602 |
| Report Service application for reporting application | 602 |
| How to Set up Azure Active Directory to perform authentication using Single Sign-On for Bold Reports On-Premise | 602 |
| Steps to set up Azure Active Directory for Bold Reports On-Premise | 602 |
| Prerequisites | 602 |
| Setup Azure Active Directory application | 603 |

| | |
|----------------------------------------------------------------------------------------------------------------|-----|
| Configure the Azure Active directory details in Bold Reports On-Premise to perform Single Sign-On | 606 |
| Configure the Azure Active directory details in Bold Reports On-Premise to import users and groups | 606 |
| How to configure Gmail SMTP Server in Email Settings | 607 |
| Gmail SMTP Server Details..... | 607 |
| Report link with filter parameters | 608 |
| How to filter data based on user in a report | 608 |
| Connecting to Data | 608 |
| Create query parameter | 610 |
| Filter data using user collection..... | 611 |
| Display data in table..... | 611 |
| Adding header to the report..... | 612 |
| Report Preview | 612 |
| How to export the report from Bold Reports Report Server..... | 613 |
| How to generate the access token for Bold Reports Report Server user | 615 |
| How to get the list of items from Bold Reports Report Server..... | 617 |
| How to apply the patches to Bold Reports Multi-tenant Azure App Service | 619 |
| How to switch the installed Bold Reports Report Server IIS Express application to IIS..... | 620 |
| How to grant access to all users for Report Server..... | 621 |
| Grant access to all users in new application..... | 621 |
| Grant access to all users in the existing application | 622 |
| Where can i find the error and debug log files | 622 |
| Error logs..... | 622 |
| Debug logs..... | 622 |
| Log Directories | 622 |
| What are the features need to be enabled in IIS to run the Bold Reports Application in Windows Client OS | 622 |
| Steps to enable the IIS and features that are needed to run the Bold Report Server in Windows Client OS | 623 |
| Required web server components..... | 623 |
| What are the features need to be enabled in IIS to run Bold Reports Application in Windows Server OS | 624 |
| Steps to enable the IIS and features that are needed to run the Bold Reports Server in Windows Server OS..... | 624 |
| Required web server components..... | 624 |
| Centralized report authoring | 625 |

| | |
|-------------------------------------------------------------------------------------------------------------|-----|
| Is Bold Reports have a centralized report authoring system..... | 625 |
| Will Syncfusion charge for Bold Reports Azure App Service..... | 625 |
| Where can i find the error and debug log files of Bold Reports Azure App Service | 625 |
| Error logs..... | 625 |
| Debug logs..... | 625 |
| Log directories..... | 625 |
| Steps to get the log files from Bold Reports Azure App service | 626 |
| Why should you migrate to Bold Reports from Syncfusion Report Platform..... | 627 |
| Is it possible to migrate from Syncfusion Report Platform Report Server to Bold Reports Report Server | 627 |
| See also | 627 |
| Where should I replace the Bold Reports license key in Bold Reports Azure App Service..... | 627 |
| See also | 627 |
| Where does the data resides in Bold Reports | 627 |
| See also | 628 |
| Does Report Server store reports execution data to the server database..... | 628 |
| See also | 628 |
| REST API | 628 |
| Overview | 628 |
| Key features | 629 |
| Overview | 629 |
| Key features | 629 |
| System Requirements | 629 |
| Hardware Requirements..... | 629 |
| Software Requirements | 630 |
| Supported Operating Systems | 630 |
| Browser Compatibility | 630 |
| See Also..... | 630 |
| Download and installation | 630 |
| Register and download Embedded Reporting Tools | 630 |
| Installing Embedded Reporting Tools | 631 |
| Embedded Reporting Tools samples and demos..... | 631 |
| View the product demos..... | 632 |
| JavaScript | 632 |
| Angular | 632 |

| | |
|-------------------------------------------------------------------------------------------------------------------------------------------|-----|
| ASP.NET CORE | 633 |
| ASP.NET MVC | 633 |
| ASP.NET Web Forms | 633 |
| WPF | 634 |
| UWP | 634 |
| Embedded Reporting Tools nuget packages and assemblies | 634 |
| Script References | 634 |
| JavaScript Embedded Reporting Tools..... | 635 |
| Angular Embedded Reporting Tools | 635 |
| ASP.NET Core Embedded Reporting Tools..... | 635 |
| Assemblies Details | 635 |
| ASP.NET MVC Embedded Reporting Tools | 635 |
| ASP.NET WebForms Embedded Reporting Tools | 636 |
| WPF Embedded Reporting Tools | 636 |
| UWP Embedded Reporting Tools..... | 636 |
| NPM Packages..... | 637 |
| Migrate Reporting Application | 640 |
| Migrate JavaScript Embedded Reporting Tools..... | 640 |
| Migrate Angular Embedded Reporting Tools | 641 |
| Migrate ASP.NET Core Embedded Reporting Tools..... | 641 |
| Migrate ASP.NET MVC Embedded Reporting Tools..... | 642 |
| Migrate ASP.NET Embedded Reporting Tools | 642 |
| Migrate WPF Embedded Reporting Tools..... | 643 |
| Migrate UWP Embedded Reporting Tools..... | 643 |
| Uninstallation of Reporting Tools | 644 |
| Frequently asked questions | 644 |
| Troubleshooting Embedded Reporting Tools | 645 |
| Controlled folder access and the protected file usage | 646 |
| Steps to allow Bold Reports Embedded Reporting Tools to access protected folders..... | 646 |
| References | 647 |
| Silent installation | 648 |
| Is Bold Reports Embedded Reporting Tools or Viewer SDK requires additional licensing when running application with Azure App service..... | 648 |
| How to provide the permission for user to access the SSRS Report Server reports | 648 |
| Site Setting of Report Server..... | 648 |

| | |
|----------------------------------------------------------------------------------------------------------|-----|
| Permission of user..... | 649 |
| Folder | 649 |
| Permission for Folder..... | 649 |
| Permission of user with Folder | 649 |
| How to use the ReportViewer along with EJ2 controls | 649 |
| Angular..... | 649 |
| JavaScript, ASP.NET Webforms, ASP.NET MVC, ASP.NET Core | 650 |
| Adding script compatibility for ASP.NET Core | 652 |
| How to use the Bold Reports with ASP.NET Core 3.x | 652 |
| What are all the SSRS versions are supported in Bold Reports | 652 |
| Centralized report authoring | 653 |
| Is Bold Reports have a centralized report authoring system..... | 653 |
| Get the user from database and pass the user as parameter for filtering the data from database | 653 |
| Does Bold Reports Embedded Reporting Tools support .NET Core and Docker on Linux..... | 654 |
| install System.Drawing native dependencies | 654 |
| How to use the Bold Reports with ASP.NET Core 3.1 | 654 |
| Does Bold Reports have any licensing procedure for deployment..... | 655 |
| Is it possible to create the RDLC reports using the business object data source in Report Designer | 655 |
| Can you use the Report Designer component from Syncfusion community license | 655 |
| How to add a header authorization for report service..... | 655 |
| Can the Bold Reports be used with ASP.NET Core on Linux and macOS..... | 655 |
| Linux..... | 656 |
| macOS | 656 |
| Is it possible to use the .NET class objects in ReportViewer..... | 656 |
| See Also..... | 656 |
| Is theme studio available for Bold Reports to generate the custom theme | 657 |
| Is it possible to create a WPF .Net Core application with Bold Report Viewer | 657 |
| Why should you migrate to Bold Reports from Syncfusion Report Platform..... | 657 |
| Is it possible to load the million records report with Report Viewer and Report Writer | 657 |
| Can specify ReportServer URL in Web API controller | 658 |
| Difference between Report Service URL and Report Server URL | 658 |
| Report Service URL..... | 658 |
| Report Server URL..... | 658 |
| How to use the Bold Reports along with EJ1 controls | 658 |
| Report Viewer SDK..... | 659 |

| | |
|--------------------------------------------------------------------------------|-----|
| Key features | 659 |
| See Also..... | 659 |
| Overview | 660 |
| Key features | 660 |
| Overview | 660 |
| Key features | 660 |
| System requirements..... | 661 |
| Hardware requirements..... | 661 |
| Supported operating systems | 661 |
| Download and installation | 661 |
| Register and download Standalone Report Designer | 661 |
| Installing Standalone Report Designer..... | 661 |
| Designing reports..... | 662 |
| Create a report..... | 663 |
| Open a report..... | 663 |
| Save a report..... | 663 |
| Preview a report | 663 |
| Uninstallation of Standalone Report Designer | 663 |
| How to queries for Bold Reports Report Designer | 663 |
| How to pass the login user as parameter to Stored Procedure..... | 664 |
| How to add interactive sorting for tablix and matrix header in the report..... | 664 |
| Frequently asked questions | 665 |
| How to do automatic resizing of Textbox based on their content | 665 |
| How to change data source for a report using the parameter..... | 665 |
| Bold reporting tools for Blazor..... | 665 |
| How to best read this user guide | 665 |
| Getting help | 665 |
| Overview | 665 |
| Add Web Report Viewer in Blazor application | 666 |
| Prerequisites | 666 |
| Create a Blazor application | 666 |
| Create a Web API for report processing | 666 |
| Initialize the Report Viewer | 669 |
| Run the Application..... | 672 |
| See Also..... | 672 |

| | |
|--------------------------------------------------------------------|-----|
| Bold reporting tools for Blazor..... | 672 |
| Key features | 672 |
| Add Web Report Designer in Blazor application..... | 673 |
| Prerequisites | 673 |
| Create a Blazor application | 673 |
| Create a Web API for report designing | 673 |
| Initialize the Report Designer | 678 |
| Run the Application..... | 681 |
| See Also..... | 681 |
| Syncfusion reporting tools for Angular | 681 |
| How to best read this user guide | 681 |
| Getting help | 681 |
| Reporting tools for Angular | 681 |
| How to best read this user guide | 681 |
| Getting help | 682 |
| System Requirements | 682 |
| Supported Operating Systems | 682 |
| Software Requirements | 682 |
| Browser Compatibility | 682 |
| See Also | 682 |
| Overview | 682 |
| Display SSRS RDL report in Bold Reports Angular Report Viewer..... | 683 |
| Prerequisites | 683 |
| Install the Angular CLI | 683 |
| Create a new application | 683 |
| Configure Bold Report Viewer in Angular CLI | 683 |
| Adding CSS reference..... | 685 |
| Adding Report Viewer component | 686 |
| Create Web API service..... | 688 |
| Adding already created report..... | 688 |
| Set report path and Web API service..... | 688 |
| Serve the application | 689 |
| See Also | 689 |
| Load SSRS Report Server reports | 690 |
| Network credentials for SSRS | 691 |

| | |
|----------------------------------------------------------|-----|
| Set data source credential for shared data sources | 691 |
| Change data source connection string | 691 |
| Render linked reports | 692 |
| Load SharePoint Server reports | 693 |
| Forms credential for SharePoint server..... | 694 |
| Set data source credential for shared data sources | 694 |
| Load Bold Report Server reports | 694 |
| Render RDLC report | 697 |
| Bind data source at client side..... | 697 |
| Bind data source in Web API controller..... | 699 |
| Load report as stream..... | 701 |
| View report click | 701 |
| Render subreport..... | 702 |
| Change subreport path | 703 |
| Set subreport parameter | 704 |
| Modify subreport data source connection string..... | 704 |
| Set subreport data source | 705 |
| Load subreport stream | 707 |
| Report parameters..... | 708 |
| Set parameter at client side..... | 708 |
| Set parameters in Web API Controller..... | 709 |
| Get report parameter | 710 |
| Report interaction events | 711 |
| Report loaded | 711 |
| Report error | 715 |
| Show error | 716 |
| Drill through..... | 717 |
| Hyperlink..... | 718 |
| Handle post actions | 719 |
| AjaxBeforeLoad | 720 |
| Add custom header in Ajax request..... | 720 |
| Pass custom data in Ajax request | 721 |
| AjaxSuccess | 724 |
| AjaxError | 725 |
| Print report | 726 |

| | |
|------------------------------------------------------------|-----|
| View report in print mode | 726 |
| Print in new page | 727 |
| Set page orientation and paper size | 728 |
| Set report margin..... | 729 |
| Set page height and width | 730 |
| Print report with images | 731 |
| External styles in report printing | 733 |
| Show print progress | 734 |
| Remove empty spaces in printing..... | 736 |
| Export report..... | 736 |
| Export event handling..... | 736 |
| Export data visualization items..... | 737 |
| Export data visualization items in azure environment | 739 |
| Change Excel and Word export format..... | 740 |
| Hide specific export type for report..... | 741 |
| PDF export options | 742 |
| Export with complex scripts..... | 742 |
| PDF conformance..... | 742 |
| Add custom PDF fonts..... | 743 |
| Word export options..... | 744 |
| Word document type..... | 744 |
| Word document advance layout for merged cells | 744 |
| Protecting Word document from editing | 744 |
| Excel export options..... | 745 |
| Excel document type..... | 745 |
| Excel document advance layout for merged cells | 745 |
| Protecting Excel document from editing | 745 |
| PowerPoint export options..... | 746 |
| CSV export options..... | 746 |
| HTML export options | 746 |
| Password protect exported document..... | 747 |
| Toolbar customization | 748 |
| Hide parameter block and toolbar items..... | 748 |
| Enable stop option in toolbar | 750 |
| Hide toolbar | 751 |

| | |
|------------------------------------------------------------|-----|
| Decide or hide the export option..... | 752 |
| Add custom items to the export drop-down | 753 |
| Add custom toolbar item | 754 |
| Add custom item to exiting toolbar group..... | 754 |
| Add new toolbar group | 756 |
| Custom actions..... | 758 |
| Send a report as an email attachment..... | 758 |
| Add email button in Report Viewer | 758 |
| Create custom email action | 760 |
| Localization of Bold Reports Angular Report Viewer..... | 761 |
| Responsive layout rendering of Angular Report Viewer | 763 |
| Normal layout | 764 |
| Responsive layout..... | 764 |
| Limitations | 764 |
| RDL specification..... | 764 |
| Report layout | 764 |
| Expressions..... | 764 |
| SSRS..... | 764 |
| Samples and demos | 765 |
| Locally installed reports | 765 |
| Offline demos..... | 765 |
| Online demos | 765 |
| GitHub demo samples..... | 765 |
| Migrate Report Viewer application | 765 |
| Server-side migration..... | 765 |
| Web API Controller | 765 |
| Report export configuration | 766 |
| NuGet Packages for Angular | 767 |
| Configure NuGet feed URL..... | 767 |
| Online NuGet feed URL..... | 767 |
| Offline NuGet feed URL..... | 768 |
| Installing NuGet packages..... | 768 |
| Install using NuGet Package Manager | 768 |
| Install using Package Manager Console | 768 |
| install specified package in default project | 768 |

| | |
|----------------------------------------------------------------------------------|-----|
| install specified package in default project with specified package source | 768 |
| install specified package in specified project..... | 768 |
| install specified package in default project | 769 |
| install specified package in default project with specified Package Source | 769 |
| install specified package in specified project..... | 769 |
| Upgrading NuGet packages | 769 |
| Upgrading using NuGet Package Manager | 769 |
| Upgrading using Package Manger Console..... | 769 |
| Update specific NuGet package in default project | 769 |
| Update all the packages in default project..... | 769 |
| Update specified package in default project with specified package source..... | 769 |
| Update specified package in specified project | 770 |
| Update specified Bold Reports NuGet package | 770 |
| Update specified package in default project with specified Package Source..... | 770 |
| Update specified package in specified project | 770 |
| Upgrading using NuGet CLI | 770 |
| update all NuGet packages from config file..... | 770 |
| update all NuGet packages from specified Packages Source | 770 |
| Update all NuGet packages from config file | 771 |
| Update all NuGet packages from specified Packages Source | 771 |
| Deployment | 771 |
| Frequently asked questions | 772 |
| Is possible to change the culture of the date time parameter | 772 |
| Is it possible to hide the parameters in Report Viewer | 772 |
| Is it possible to hide the export options in Report Viewer | 772 |
| Is it possible to load reports providing parameter values at runtime | 773 |
| Angular Report Viewer Reporting Service | 773 |
| IReportController | 773 |
| Create ASP.NET Web API service | 774 |
| Configure Report Viewer Web API..... | 775 |
| ReportHelper..... | 775 |
| Add Web API Controller..... | 775 |
| Add routing information | 777 |
| Enable Cross-Origin Requests | 777 |
| Create ASP.NET Core Web API Service | 779 |

| | |
|---------------------------------------------------------------------------|-----|
| List of dependency Libraries | 779 |
| Configure Web API..... | 780 |
| Add Web API Controller..... | 780 |
| Enable Cross-Origin requests | 783 |
| How to section for Angular Report Viewer..... | 784 |
| Display SSRS RDL report in Bold Reports Angular Report Viewer..... | 784 |
| Prerequisites | 784 |
| Install the Angular CLI | 784 |
| Create a new application | 785 |
| Configure Bold Report Viewer in Angular CLI | 785 |
| Adding CSS reference..... | 786 |
| Adding Report Viewer component | 787 |
| Create Web API service..... | 789 |
| Adding already created report..... | 789 |
| Set report path and Web API service..... | 790 |
| Serve the application | 790 |
| See Also..... | 791 |
| Create a SSRS RDL report..... | 791 |
| Bold Reports Report Designer | 791 |
| Microsoft SQL Report Builder | 791 |
| Visual Studio Report Server template..... | 791 |
| Create a RDLC report using business object data source | 791 |
| Prerequisites | 792 |
| Create business object class | 792 |
| Add an RDLC report..... | 792 |
| Data source and table configuration wizard..... | 793 |
| How to change the exporting document file name based on parameter | 793 |
| How to change the connection string datasource dynamically..... | 794 |
| How to disable the vertical scrollbar in parameter panel | 795 |
| Reporting tools for Angular | 796 |
| Key features | 796 |
| Add Web Report Designer to an Angular application using Angular CLI | 796 |
| Prerequisites | 796 |
| Install the Angular CLI | 797 |
| Create a new application | 797 |

| | |
|-----------------------------------------------------|-----|
| Configure Bold Report Designer in Angular CLI | 797 |
| Adding CSS reference..... | 799 |
| Adding code mirror reference | 800 |
| Adding Scripts reference..... | 802 |
| Adding Report Designer component | 804 |
| Create Web API service..... | 806 |
| Set Web API service URL..... | 806 |
| Serve the application | 807 |
| See Also..... | 807 |
| Localization | 807 |
| Integrate the component with Report Server | 809 |
| Designing Reports | 812 |
| Connecting your data..... | 812 |
| Transform your data | 812 |
| Working with report parameters..... | 812 |
| Embed images into the report..... | 812 |
| Interacting with report design surface | 813 |
| Report items | 813 |
| Customize appearance..... | 813 |
| Report design settings | 813 |
| Shape report data | 813 |
| Interactive features..... | 813 |
| Working with Expressions..... | 813 |
| Open report | 813 |
| Save report..... | 813 |
| Preview report | 813 |
| Migrate Report Designer application..... | 813 |
| NPM packages..... | 814 |
| Server-side migration..... | 817 |
| Assemblies | 817 |
| Packages..... | 817 |
| Namespace changes | 817 |
| Control initialization..... | 818 |
| Report export configuration | 818 |
| NuGet Packages for Angular | 819 |

| | |
|----------------------------------------------------------------------------------|-----|
| Configure NuGet feed URL..... | 819 |
| Online NuGet feed URL..... | 819 |
| Offline NuGet feed URL..... | 820 |
| Installing NuGet packages..... | 820 |
| Install using NuGet Package Manager | 820 |
| Install using Package Manager Console | 820 |
| install specified package in default project | 821 |
| install specified package in default project with specified package source | 821 |
| install specified package in specified project..... | 821 |
| install specified package in default project | 821 |
| install specified package in default project with specified Package Source | 821 |
| install specified package in specified project..... | 821 |
| Upgrading NuGet packages | 821 |
| Upgrading using NuGet Package Manager | 821 |
| Upgrading using Package Manger Console..... | 821 |
| Update specific NuGet package in default project | 822 |
| Update all the packages in default project..... | 822 |
| Update specified package in default project with specified package source..... | 822 |
| Update specified package in specified project | 822 |
| Update specified Bold Reporting NuGet package | 822 |
| Update specified package in default project with specified Package Source..... | 822 |
| Update specified package in specified project | 822 |
| Upgrading using NuGet CLI | 822 |
| update all NuGet packages from config file..... | 823 |
| update all NuGet packages from specified Packages Source | 823 |
| Update all NuGet packages from config file | 823 |
| Update all NuGet packages from specified Packages Source | 823 |
| Deployment | 823 |
| Responsive layout rendering of Angular Report Designer..... | 824 |
| Normal layout | 824 |
| Responsive layout..... | 824 |
| Samples and demos | 824 |
| Locally installed reports | 824 |
| Offline demos..... | 825 |
| Online demos | 825 |

| | |
|-----------------------------------------------------------------------------------|-----|
| GitHub demo samples..... | 825 |
| Supported Browsers..... | 825 |
| Angular Report Designer Reporting Service | 825 |
| IReportDesignerController..... | 825 |
| ReportDesignerHelper | 826 |
| ReportHelper..... | 826 |
| Create ASP.NET Web API Service..... | 829 |
| List of dependency Libraries | 830 |
| Add Routing Information | 831 |
| Add Web API Controller..... | 832 |
| Configure Web API..... | 832 |
| ReportDesignerHelper | 832 |
| ReportHelper..... | 833 |
| Create ASP.NET Core Web API Service | 837 |
| List of dependency Libraries | 837 |
| Register CORS..... | 838 |
| Add API Service | 838 |
| Configure Web API..... | 838 |
| ReportDesignerHelper | 839 |
| ReportHelper..... | 839 |
| How to section for Angular Report Designer..... | 843 |
| Add Web Report Designer to an Angular application using Angular CLI | 843 |
| Prerequisites | 843 |
| Install the Angular CLI | 844 |
| Create a new application | 844 |
| Configure Bold Report Designer in Angular CLI | 844 |
| Adding CSS reference..... | 846 |
| Adding code mirror reference | 847 |
| Adding Scripts reference..... | 849 |
| Adding Report Designer component | 851 |
| Create Web API service..... | 852 |
| Set Web API service URL..... | 853 |
| Serve the application | 853 |
| See Also..... | 854 |
| How to add the data source and dataset for Report Designer from application | 854 |

| | |
|-----------------------------------------------------------------------------|-----|
| Breaking Changes..... | 858 |
| Breaking Changes..... | 858 |
| Designer resource read and write API..... | 858 |
| Parameters..... | 858 |
| Return Type..... | 859 |
| Code snippet | 859 |
| Breaking Changes..... | 860 |
| Breaking Changes..... | 860 |
| Reporting tools for JavaScript..... | 861 |
| How to best read this user guide | 861 |
| Getting help | 861 |
| Reporting tools for JavaScript..... | 861 |
| How to best read this user guide | 861 |
| Getting help | 862 |
| System Requirements | 862 |
| Supported Operating Systems | 862 |
| Hardware Requirements..... | 862 |
| Software Requirements | 862 |
| Browser Compatibility | 862 |
| See Also | 862 |
| Overview | 862 |
| Display ssrs rdl report in Bold Reports HTML5 JavaScript Report Viewer..... | 863 |
| HTML file creation..... | 863 |
| Refer scripts and CSS..... | 863 |
| Adding Report Viewer Widget | 864 |
| Create Web API service..... | 865 |
| Adding already created report..... | 865 |
| Set report path and Web API service..... | 865 |
| Preview the report..... | 866 |
| See Also | 867 |
| Load SSRS Report Server reports | 868 |
| Network credentials for SSRS | 868 |
| Set data source credential for shared data sources | 869 |
| Change data source connection string | 870 |
| Render linked reports | 870 |

| | |
|----------------------------------------------------------|-----|
| Load SharePoint Server reports | 871 |
| Forms credential for SharePoint server | 871 |
| Set data source credential for shared data sources | 872 |
| Load Bold Report Server reports | 873 |
| Render RDLC report | 875 |
| Bind data source at client side | 875 |
| Bind data source in Web API controller | 877 |
| Load report as stream | 879 |
| View Report Click | 879 |
| Render subreport | 880 |
| Change subreport path | 881 |
| Set subreport parameter | 881 |
| Modify subreport data source connection string | 882 |
| Set subreport data source | 882 |
| Load subreport stream | 885 |
| Report parameters | 885 |
| Set parameter at client | 886 |
| Set parameters in Web API Controller | 887 |
| Get report parameter | 887 |
| Report interaction events | 888 |
| Report loaded | 888 |
| Report error | 889 |
| Show error | 889 |
| Drill through | 890 |
| Hyperlink | 890 |
| Handle post actions | 891 |
| AjaxBeforeLoad | 891 |
| Add custom header in Ajax request | 891 |
| Pass custom data in Ajax request | 892 |
| AjaxSuccess | 894 |
| AjaxError | 895 |
| Error logging in JavaScript Report Viewer | 895 |
| Print report | 899 |
| View report in print mode | 899 |
| Print in new page | 899 |

| | |
|------------------------------------------------------------|-----|
| Set page orientation and paper size | 900 |
| Set report margin..... | 900 |
| Set page height and width | 900 |
| Print report with images | 901 |
| External styles in report printing | 902 |
| Show print progress | 902 |
| Remove empty spaces in printing..... | 903 |
| Export report..... | 903 |
| Export event handling..... | 903 |
| Export data visualization items..... | 904 |
| Export data visualization items in azure environment | 905 |
| Change Excel and Word export format..... | 907 |
| Hide specific export type for report..... | 907 |
| PDF export options | 907 |
| Export with complex scripts..... | 907 |
| PDF Conformance | 908 |
| Add custom PDF fonts..... | 908 |
| Word export options..... | 909 |
| Word document type..... | 909 |
| Word document advance layout for merged cells | 909 |
| Protecting Word document from editing | 910 |
| Excel export options..... | 910 |
| Excel document type..... | 910 |
| Excel document advance layout for merged cells | 911 |
| Protecting Excel document from editing | 911 |
| PowerPoint export options..... | 911 |
| CSV export options..... | 911 |
| HTML export options | 912 |
| Password protect exported document..... | 912 |
| Toolbar customization | 913 |
| Hide parameter block and toolbar items..... | 913 |
| Enable stop option in toolbar | 914 |
| Hide toolbar | 914 |
| Decide or hide the export option..... | 915 |
| Add custom items to the export drop-down | 915 |

| | |
|---------------------------------------------------------------------|-----|
| Add custom toolbar item | 916 |
| Add custom item to exiting toolbar group..... | 916 |
| Add new toolbar group | 916 |
| Custom Actions | 917 |
| Send a report as an email attachment..... | 917 |
| Add email button in Report Viewer | 917 |
| Create custom email action | 918 |
| Cancel report processing in Report Viewer | 920 |
| Add cancel report button..... | 920 |
| Localization of Bold Reports HTML5 JavaScript Report Viewer..... | 922 |
| Responsive layout rendering of HTML5 JavaScript Report Viewer | 923 |
| Normal layout | 924 |
| Responsive layout..... | 924 |
| Limitations | 924 |
| RDL Specification..... | 924 |
| Report Layout..... | 924 |
| Expressions..... | 924 |
| SSRS..... | 924 |
| Samples and demos | 924 |
| Locally installed reports | 924 |
| Offline demos..... | 925 |
| Online demos | 925 |
| GitHub demo samples..... | 925 |
| Reporting Service application | 925 |
| Migrate Report Viewer application | 925 |
| Client-side migration..... | 925 |
| Adding data visualization scripts..... | 926 |
| Server-side migration..... | 927 |
| Web API Controller | 927 |
| Report export configuration | 927 |
| NuGet Packages for JavaScript..... | 929 |
| Configure NuGet feed URL..... | 929 |
| Online NuGet feed URL..... | 929 |
| Offline NuGet feed URL..... | 929 |
| Installing NuGet packages..... | 930 |

| | |
|----------------------------------------------------------------------------------|-----|
| Install using NuGet Package Manager | 930 |
| Install using Package Manager Console | 930 |
| install specified package in default project | 930 |
| install specified package in default project with specified package source | 930 |
| install specified package in specified project..... | 930 |
| install specified package in default project | 931 |
| install specified package in default project with specified Package Source | 931 |
| install specified package in specified project..... | 931 |
| Upgrading NuGet packages | 931 |
| Upgrading using NuGet Package Manager | 931 |
| Upgrading using Package Manger Console..... | 931 |
| Update specific NuGet package in default project | 931 |
| Update all the packages in default project..... | 931 |
| Update specified package in default project with specified package source..... | 932 |
| Update specified package in specified project | 932 |
| Update specified Bold Reporting NuGet package | 932 |
| Update specified package in default project with specified Package Source..... | 932 |
| Update specified package in specified project | 932 |
| Upgrading using NuGet CLI | 932 |
| update all NuGet packages from config file..... | 932 |
| update all NuGet packages from specified Packages Source | 933 |
| Update all NuGet packages from config file | 933 |
| Update all NuGet packages from specified Packages Source | 933 |
| CDN | 933 |
| CDN scripts links..... | 933 |
| Bold Reports dependency libraries | 933 |
| External dependency libraries | 936 |
| CDN style sheet links..... | 937 |
| Refer local Scripts and CSS when CDN fails | 938 |
| JavaScript Report Viewer Reporting Service..... | 940 |
| IReportController | 940 |
| Create ASP.NET Web API service | 941 |
| Configure Report Viewer Web API..... | 942 |
| ReportHelper..... | 942 |
| Add Web API Controller..... | 942 |

| | |
|-----------------------------------------------------------------------------------|-----|
| Add routing information | 944 |
| Enable Cross-Origin Requests | 944 |
| Create ASP.NET Core Web API Service | 946 |
| List of dependency Libraries | 946 |
| Configure Web API..... | 947 |
| Add Web API Controller | 947 |
| Enable Cross-Origin requests..... | 950 |
| How to queries for Bold Reports ReportViewer | 951 |
| Create a SSRS RDL report..... | 951 |
| Bold Reports Report Designer | 952 |
| Microsoft SQL Report Builder | 952 |
| Visual Studio Report Server template..... | 952 |
| Create a RDLC report using business object data source | 952 |
| Prerequisites | 952 |
| Create business object class | 952 |
| Add an RDLC report..... | 953 |
| Data source and table configuration wizard..... | 953 |
| Adding data visualization scripts..... | 954 |
| How to use JavaScript ReportViewer in `Blazor Web Assembly` App application | 955 |
| Refer a scripts and CSS..... | 955 |
| Adding a ReportViewer | 956 |
| Create a Web API service | 957 |
| How to use Bold Reports JavaScript ReportViewer with `Blazor Server` App | 959 |
| Refer a scripts and CSS..... | 959 |
| Adding a ReportViewer | 960 |
| Create a Web API service | 961 |
| How to change the exporting document file name based on parameter | 963 |
| How to change the connection string datasource dynamically..... | 963 |
| How to disable the vertical scrollbar in parameter panel | 965 |
| JavaScript Report Viewer API reference | 965 |
| Members..... | 965 |
| Methods..... | 985 |
| destroy()..... | 985 |
| exportReport() | 985 |
| fitToPage()..... | 985 |

| | |
|-------------------------------------|------|
| fitToPageHeight() | 986 |
| fitToPageWidth() | 986 |
| getDataSetNames() | 986 |
| getParameters() | 986 |
| gotoFirstPage() | 987 |
| gotoLastPage() | 987 |
| gotoNextPage() | 987 |
| gotoPageIndex() | 988 |
| gotoPreviousPage() | 988 |
| print() | 988 |
| printLayout() | 988 |
| refresh() | 989 |
| setParameterBlockVisibility() | 989 |
| cancelRendering() | 989 |
| Events | 990 |
| drillThrough | 990 |
| renderingBegin | 990 |
| renderingComplete | 991 |
| reportError | 992 |
| reportExport | 992 |
| reportLoaded | 993 |
| showError | 993 |
| viewReportClick | 994 |
| ajaxBeforeLoad | 995 |
| ajaxSuccess | 995 |
| ajaxError | 996 |
| toolbarRendering | 996 |
| exportProgressChanged | 997 |
| printProgressChanged | 997 |
| exportItemClick | 998 |
| toolBarItemClick | 999 |
| hyperlink | 999 |
| reportPrint | 1000 |
| JavaScript Report Viewer Properties | 1001 |
| pageSettings | 1001 |

| | |
|-------------------------------------------------------------------------------------------------|------|
| Properties..... | 1001 |
| parameters `array` | 1005 |
| Properties..... | 1005 |
| exportSettings..... | 1010 |
| Properties..... | 1010 |
| toolbarSettings..... | 1015 |
| Properties..... | 1015 |
| parameterSettings | 1021 |
| Properties..... | 1021 |
| dataSources `array` | 1027 |
| Properties..... | 1027 |
| Frequently asked questions | 1028 |
| How can improve the performance and handle the large amounts of data with Report Viewer..... | 1029 |
| Is Report Viewer compatible with latest version of JQuery library | 1029 |
| Is the back action from drillthrough report will load the report again with Report Viewer | 1029 |
| Is possible to change the culture of the date time parameter | 1029 |
| Is it possible to hide the parameters in Report Viewer | 1030 |
| Is it possible to hide the export options in Report Viewer | 1030 |
| Is it possible to load reports providing parameter values at runtime | 1030 |
| Reporting tools for JavaScript | 1030 |
| Key features | 1030 |
| Add Web Report Designer to a javascript application..... | 1030 |
| HTML file creation..... | 1030 |
| Refer Scripts and CSS | 1031 |
| Initialize Report Designer..... | 1032 |
| Create Web API service..... | 1033 |
| Set service URL..... | 1033 |
| Run the application..... | 1033 |
| Dependencies..... | 1039 |
| Scripts..... | 1039 |
| Styles | 1044 |
| Localization | 1046 |
| Integrate the component with Report Server | 1052 |
| Run the application..... | 1054 |
| Designing Reports | 1055 |

| | |
|----------------------------------------------|------|
| Connecting your data..... | 1055 |
| Transform your data | 1056 |
| Working with report parameters..... | 1056 |
| Embed images into the report..... | 1056 |
| Interacting with report design surface | 1056 |
| Report items | 1056 |
| Customize appearance..... | 1056 |
| Report design settings | 1056 |
| Shape report data | 1056 |
| Interactive features..... | 1057 |
| Working with Expressions..... | 1057 |
| Open report | 1057 |
| Save report..... | 1057 |
| Preview report | 1057 |
| Samples and demos | 1057 |
| Offline demos..... | 1057 |
| Online demos | 1057 |
| GitHub demo samples..... | 1057 |
| Reporting Service application | 1057 |
| Migrate Report Designer application..... | 1057 |
| Client-side migration..... | 1058 |
| Scripts..... | 1058 |
| Styles | 1059 |
| Server-side migration..... | 1059 |
| Assemblies | 1059 |
| Packages..... | 1060 |
| Namespace changes | 1060 |
| Control initialization..... | 1060 |
| Report export configuration | 1060 |
| NuGet Packages for JavaScript..... | 1062 |
| Configure NuGet feed URL..... | 1062 |
| Online NuGet feed URL..... | 1062 |
| Offline NuGet feed URL..... | 1062 |
| Installing NuGet Packages..... | 1063 |
| Install using NuGet Package Manager | 1063 |

| | |
|----------------------------------------------------------------------------------|------|
| Install using Package Manager Console | 1063 |
| install specified package in default project | 1063 |
| install specified package in default project with specified package source | 1063 |
| install specified package in specified project..... | 1063 |
| install specified package in default project | 1064 |
| install specified package in default project with specified Package Source | 1064 |
| install specified package in specified project..... | 1064 |
| Upgrading NuGet packages | 1064 |
| Upgrade using NuGet Package Manager | 1064 |
| Upgrade using Package Manger Console..... | 1064 |
| Update specific NuGet package in default project | 1064 |
| Update all the packages in default project..... | 1064 |
| Update specified package in default project with specified package source..... | 1065 |
| Update specified package in specified project | 1065 |
| Update specified Bold Reporting NuGet package | 1065 |
| Update specified package in default project with specified Package Source..... | 1065 |
| Update specified package in specified project | 1065 |
| Upgrade using NuGet CLI | 1065 |
| update all NuGet packages from config file..... | 1065 |
| update all NuGet packages from specified Packages Source | 1066 |
| Update all NuGet packages from config file | 1066 |
| Update all NuGet packages from specified Packages Source | 1066 |
| Responsive layout rendering of HTML5 JavaScript Report Designer..... | 1066 |
| Normal layout | 1066 |
| Responsive layout | 1066 |
| Supported Browsers..... | 1066 |
| Accessibility..... | 1067 |
| Javascript Report Designer Reporting Service | 1067 |
| IReportDesignerController..... | 1067 |
| ReportDesignerHelper | 1068 |
| ReportHelper..... | 1068 |
| Create ASP.NET Web API Service..... | 1071 |
| List of dependency libraries | 1072 |
| Add Routing Information | 1073 |
| Add Web API Controller..... | 1074 |

| | |
|---------------------------------------------------------------------------------------------|------|
| Configure Web API..... | 1074 |
| ReportDesignerHelper | 1074 |
| ReportHelper..... | 1075 |
| Create ASP.NET Core Web API Service | 1078 |
| List of dependency Libraries | 1079 |
| Register CORS..... | 1080 |
| Add API Service | 1080 |
| Configure Web API..... | 1080 |
| ReportDesignerHelper | 1081 |
| ReportHelper..... | 1081 |
| How to queries for Bold Reports Report Designer | 1085 |
| Create a RDLC report | 1085 |
| Create an application | 1085 |
| Set report type property..... | 1085 |
| Create data..... | 1086 |
| Assign data | 1087 |
| Preview report | 1087 |
| See Also..... | 1089 |
| How to open server report using report path at the time of opening the Report Designer..... | 1089 |
| How to add the data source and dataset for Report Designer from application | 1090 |
| Breaking Changes..... | 1094 |
| Breaking Changes..... | 1094 |
| Designer resource read and write API | 1094 |
| Parameters..... | 1094 |
| Return Type..... | 1095 |
| Code snippet | 1096 |
| JavaScript Report Designer API reference | 1096 |
| Members..... | 1096 |
| Properties..... | 1096 |
| Methods..... | 1105 |
| addDataSet..... | 1105 |
| addDataSource..... | 1108 |
| addItem..... | 1110 |
| bringForward..... | 1116 |
| bringToFront | 1116 |

| | |
|-----------------------------|------|
| canCopy..... | 1117 |
| canCut | 1117 |
| canPaste..... | 1117 |
| canRedo | 1118 |
| canRemove..... | 1118 |
| canUndo..... | 1119 |
| cloneDataSet..... | 1119 |
| cloneDataSource..... | 1119 |
| copy..... | 1120 |
| cut | 1120 |
| hasReportChanges..... | 1121 |
| isNewReport | 1121 |
| isNewServerReport | 1121 |
| isServerReport..... | 1122 |
| newReport | 1122 |
| newServerReport | 1123 |
| openReport | 1123 |
| openReportDefinition | 1124 |
| openReportFromDevice..... | 1124 |
| openServerReportDialog..... | 1124 |
| paste..... | 1125 |
| redo..... | 1125 |
| remove | 1125 |
| removeDataSet | 1126 |
| removeDatasource..... | 1126 |
| removeItem | 1127 |
| saveReport | 1127 |
| saveReportDefinition | 1128 |
| saveServerReportDialog..... | 1129 |
| saveToDevice | 1129 |
| selectItems..... | 1129 |
| sendBackward | 1130 |
| sendToBack | 1130 |
| showDesign..... | 1131 |
| showNewReportDialog | 1131 |

| | |
|---------------------------------------------|------|
| showOpenSaveReportDialog | 1131 |
| showPreview..... | 1133 |
| undo | 1133 |
| updateDataset | 1133 |
| updateDatasource..... | 1137 |
| Events..... | 1139 |
| ajaxBeforeLoad | 1139 |
| ajaxError..... | 1140 |
| ajaxSuccess..... | 1140 |
| create | 1141 |
| destroy | 1141 |
| newDataClick | 1142 |
| openReportClick..... | 1142 |
| previewReport | 1142 |
| reportModified | 1143 |
| reportOpened | 1143 |
| reportSaved..... | 1144 |
| saveReportClick..... | 1144 |
| toolbarClick | 1145 |
| toolbarRendering | 1145 |
| JavaScript Report Designer Properties | 1146 |
| configurePaneSettings | 1146 |
| Properties..... | 1146 |
| reportDataExtensions `array` | 1149 |
| Properties..... | 1149 |
| reportItemExtensions `array` | 1152 |
| Properties..... | 1152 |
| toolbarSettings..... | 1156 |
| Properties..... | 1156 |
| pageSettings..... | 1160 |
| Properties..... | 1160 |
| previewOptions..... | 1165 |
| Properties..... | 1165 |
| Events..... | 1172 |
| renderingBegin..... | 1172 |

| | |
|----------------------------------------------------|------|
| renderingComplete | 1173 |
| reportError | 1173 |
| reportPrint | 1174 |
| parameters..... | 1175 |
| Properties..... | 1175 |
| exportSettings..... | 1179 |
| Properties..... | 1179 |
| toolbarSettings..... | 1185 |
| Properties..... | 1185 |
| parameterSettings | 1191 |
| Properties..... | 1191 |
| dataSources `array` | 1195 |
| Properties..... | 1195 |
| Available chart types..... | 1197 |
| permissionSettings..... | 1197 |
| Properties..... | 1197 |
| Bold reporting tools for React..... | 1201 |
| How to best read this user guide | 1201 |
| Getting help | 1201 |
| Bold reporting tools for React..... | 1202 |
| How to best read this user guide | 1202 |
| Getting help | 1202 |
| System Requirements | 1202 |
| Supported Operating Systems | 1202 |
| Software Requirements | 1202 |
| Browser Compatibility | 1202 |
| See Also | 1203 |
| Overview | 1203 |
| Add Web Report Viewer to a React application | 1203 |
| Prerequisites | 1203 |
| Install the Create React App package | 1203 |
| Create a new React application | 1203 |
| Install the Create React Class..... | 1204 |
| Install the Bold Reports React package..... | 1204 |
| Adding scripts reference | 1204 |

| | |
|---------------------------------------------------------------|------|
| Adding Report Viewer component | 1205 |
| Create a Web API service..... | 1206 |
| Set a Web API service URL | 1206 |
| Run the Application..... | 1207 |
| Deploying the application in production | 1207 |
| Add Web Report Viewer with Typescript React application..... | 1207 |
| Prerequisites | 1207 |
| Install the Create React App Package | 1208 |
| Create a new React Typescript Application | 1208 |
| Install Create React Class | 1208 |
| Install Bold Reports React package | 1208 |
| Install JQuery package | 1208 |
| Adding Scripts reference..... | 1209 |
| Adding Report Viewer component | 1209 |
| Create a Web API service..... | 1211 |
| Set Web API service URL..... | 1211 |
| Run the Application..... | 1212 |
| Deploying the application in production | 1212 |
| Load SSRS Report Server reports | 1212 |
| Network credentials for SSRS | 1213 |
| Set data source credential for shared data sources | 1213 |
| Change data source connection string | 1214 |
| Render linked reports | 1214 |
| Add Web Report Viewer to a React Boilerplate application..... | 1215 |
| Prerequisites | 1215 |
| React Boilerplate Application | 1215 |
| Install the Bold Reports React package..... | 1215 |
| Adding global scripts reference | 1216 |
| Configuring webpack | 1217 |
| Adding Report Viewer component | 1217 |
| Create a Web API service..... | 1218 |
| Set a Web API service URL | 1219 |
| Run the Application..... | 1220 |
| Deploying the application in production | 1220 |
| Add Web Report Viewer to a React Boilerplate application..... | 1220 |

| | |
|-----------------------------------------------------|------|
| Prerequisites | 1220 |
| React Boilerplate TypeScript Application | 1221 |
| Install the Bold Reports React package..... | 1221 |
| Adding global scripts reference | 1221 |
| Configuring webpack | 1222 |
| Adding Report Viewer component | 1224 |
| Create a Web API service..... | 1225 |
| Set a Web API service URL | 1225 |
| Run the Application..... | 1226 |
| Deploying the application in production | 1226 |
| Load Bold Reports Report Server reports..... | 1227 |
| Render RDLC report | 1229 |
| Bind data source at client side..... | 1229 |
| Bind data source in Web API controller..... | 1230 |
| Load report as stream..... | 1232 |
| View report click | 1233 |
| Render subreport..... | 1234 |
| Change subreport path | 1235 |
| Set subreport parameter | 1235 |
| Modify subreport data source connection string..... | 1235 |
| Set subreport data source | 1236 |
| Load subreport stream | 1238 |
| Report parameters..... | 1239 |
| Set parameter at client side..... | 1239 |
| Set parameters in Web API Controller..... | 1240 |
| Get report parameter | 1241 |
| Report interaction events | 1242 |
| Report loaded | 1242 |
| Report error | 1245 |
| Show error | 1246 |
| Drill through..... | 1247 |
| Hyperlink..... | 1247 |
| Handle post actions | 1248 |
| AjaxBeforeLoad | 1248 |
| Add custom header in Ajax request..... | 1248 |

| | |
|------------------------------------------------------------|------|
| Pass custom data in Ajax request | 1250 |
| AjaxSuccess | 1251 |
| AjaxError | 1252 |
| Print report | 1253 |
| View report in print mode | 1253 |
| Print in new page | 1253 |
| Set page orientation and paper size | 1254 |
| Set report margin..... | 1254 |
| Set page height and width..... | 1255 |
| Print report with images | 1256 |
| External styles in report printing | 1257 |
| Show print progress | 1257 |
| Remove empty spaces in printing..... | 1258 |
| Export report..... | 1258 |
| Export event handling..... | 1259 |
| Export data visualization items..... | 1260 |
| Export data visualization items in azure environment | 1261 |
| Change Excel and Word export format..... | 1263 |
| Hide specific export type for report..... | 1263 |
| PDF export options | 1264 |
| Export with complex scripts..... | 1264 |
| PDF conformance..... | 1264 |
| Add custom PDF fonts..... | 1264 |
| Word export options..... | 1265 |
| Word document type..... | 1265 |
| Word document advance layout for merged cells | 1266 |
| Protecting Word document from editing | 1266 |
| Excel export options..... | 1267 |
| Excel document type..... | 1267 |
| Excel document advance layout for merged cells | 1267 |
| Protecting Excel document from editing | 1267 |
| PowerPoint export options..... | 1267 |
| CSV export options..... | 1268 |
| HTML export options | 1268 |
| Password protect exported document..... | 1268 |

| | |
|------------------------------------------------------------------|------|
| Toolbar customization | 1269 |
| Hide parameter block and toolbar items..... | 1270 |
| Enable stop option in toolbar | 1271 |
| Hide toolbar | 1271 |
| Decide or hide the export option..... | 1272 |
| Add custom items to the export drop-down | 1273 |
| Add custom toolbar item | 1274 |
| Add custom item to exiting toolbar group..... | 1274 |
| Add new toolbar group | 1275 |
| Custom actions..... | 1277 |
| Send a report as an email attachment..... | 1277 |
| Add email button in Report Viewer | 1277 |
| Create custom email action | 1278 |
| Responsive layout rendering of React Report Viewer | 1280 |
| Normal layout | 1281 |
| Responsive layout | 1281 |
| Localization of Bold Reports React Report Viewer | 1281 |
| React Report Viewer Reporting Service..... | 1282 |
| IReportController | 1282 |
| Create ASP.NET Web API service | 1284 |
| Configure Report Viewer Web API..... | 1284 |
| ReportHelper..... | 1285 |
| Add Web API Controller..... | 1285 |
| Add routing information | 1286 |
| Enable Cross-Origin Requests | 1287 |
| Create ASP.NET Core Web API Service | 1289 |
| List of dependency Libraries | 1289 |
| Configure Web API | 1290 |
| Add Web API Controller..... | 1290 |
| Enable Cross-Origin requests | 1293 |
| Frequently asked questions | 1294 |
| Is it possible to hide the parameters in Report Viewer | 1294 |
| Is it possible to hide the export options in Report Viewer | 1294 |
| Bold reporting tools for React..... | 1294 |
| Key features | 1294 |

| | |
|-----------------------------------------------------------------|------|
| Add Web Report Designer to a React application | 1295 |
| Prerequisites | 1295 |
| Install the Create React App package | 1295 |
| Create a new React application | 1295 |
| Install the Create React Class | 1295 |
| Install the Bold Reports React package | 1295 |
| Adding scripts reference | 1296 |
| Adding Report Designer component | 1296 |
| Create a Web API service | 1298 |
| Set a Web API service URL | 1298 |
| Run the Application | 1299 |
| Deploying the application in production | 1299 |
| Add Web Report Designer with Typescript React application..... | 1299 |
| Prerequisites | 1299 |
| Install the Create React App Package | 1300 |
| Create a new React Typescript Application | 1300 |
| Install Create React Class | 1300 |
| Install Bold Reports React package | 1300 |
| Install JQuery package | 1300 |
| Adding Scripts reference | 1301 |
| Adding Report Designer component | 1301 |
| Create Web API service | 1303 |
| Set Web API service URL | 1303 |
| Run the Application | 1304 |
| Deploying the application in production | 1304 |
| Add Web Report Designer to a React Boilerplate application..... | 1304 |
| Prerequisites | 1304 |
| React Boilerplate Application | 1305 |
| Install the Bold Reports React package | 1305 |
| Adding global scripts reference | 1305 |
| Configuring webpack | 1306 |
| Adding Report Designer component | 1307 |
| Create a Web API service | 1308 |
| Set a Web API service URL | 1308 |
| Run the Application | 1309 |

| | |
|---------------------------------------------------------------------------------------------|------|
| Deploying the application in production | 1310 |
| Add Web Report Designer to a React Boilerplate application..... | 1310 |
| Prerequisites | 1310 |
| React Boilerplate TypeScript Application | 1310 |
| Install the Bold Reports React package..... | 1311 |
| Adding global scripts reference | 1311 |
| Configuring webpack | 1312 |
| Adding Report Designer component | 1313 |
| Create a Web API service..... | 1315 |
| Set a Web API service URL | 1315 |
| Run the Application..... | 1316 |
| Deploying the application in production | 1316 |
| Load Bold Reports Report Server reports | 1316 |
| React Report Designer Reporting Service..... | 1319 |
| IReportDesignerController..... | 1319 |
| ReportDesignerHelper | 1320 |
| ReportHelper..... | 1320 |
| Create ASP.NET Web API Service..... | 1323 |
| List of dependency libraries..... | 1324 |
| Add Routing Information | 1325 |
| Add Web API Controller..... | 1326 |
| Configure Web API..... | 1326 |
| ReportDesignerHelper | 1326 |
| ReportHelper..... | 1327 |
| Create ASP.NET Core Web API Service | 1330 |
| List of dependency Libraries | 1331 |
| Register CORS..... | 1332 |
| Add API Service | 1332 |
| Configure Web API..... | 1332 |
| ReportDesignerHelper | 1333 |
| ReportHelper..... | 1333 |
| How to queries for Bold Reports Report Designer | 1337 |
| How to open server report using report path at the time of opening the Report Designer..... | 1337 |
| How to add the data source and dataset for Report Designer from application | 1338 |
| Breaking Changes..... | 1342 |

| | |
|------------------------------------------------------------------------------------------------------------|------|
| Breaking Changes..... | 1342 |
| Designer resource read and write API | 1342 |
| Parameters..... | 1342 |
| Return Type..... | 1343 |
| Code snippet | 1344 |
| How to section for React Bold Reporting Tools | 1344 |
| How to resolve the Javascript memory heap out issue that occurs while building the React application | 1344 |
| Reporting tools for ASP.NET Core | 1345 |
| How to best read this user guide | 1345 |
| Getting help | 1345 |
| Reporting tools for ASP.NET Core | 1345 |
| How to best read this user guide | 1345 |
| Getting help | 1345 |
| System Requirements | 1345 |
| Supported Operating Systems | 1345 |
| Framework | 1346 |
| Hardware Requirements..... | 1346 |
| Software Requirements | 1346 |
| Browser Compatibility | 1346 |
| See Also | 1346 |
| Overview | 1346 |
| Display ssrs rdl report in Bold Reports ASP.NET Core Report Viewer | 1347 |
| Create ASP.NET Core application..... | 1347 |
| List of dependency libraries | 1347 |
| Refer scripts and CSS..... | 1348 |
| Tag helper | 1349 |
| Configure Script Manager | 1349 |
| Initialize Report Viewer..... | 1350 |
| Add already created reports | 1350 |
| Configure Web API..... | 1350 |
| Add Web API Controller | 1350 |
| Enable cross-origin requests | 1354 |
| Set report path and service URL | 1354 |
| Preview the report | 1355 |

| | |
|-----------------------------------------------------------|------|
| See Also..... | 1355 |
| Display SSRS rdl report in ASP.NET Core Razor Pages | 1355 |
| Prerequisites | 1355 |
| Create ASP.NET Core application..... | 1356 |
| List of dependency libraries..... | 1356 |
| Refer scripts and CSS..... | 1356 |
| Tag helper | 1358 |
| Configure Script Manager | 1358 |
| Initialize Report Viewer..... | 1358 |
| Add already created reports | 1359 |
| Configure Web API..... | 1359 |
| Add Web API Controller | 1359 |
| Web API with ASP.NET Core Web Application..... | 1362 |
| Set report path and service URL | 1363 |
| Preview the report..... | 1363 |
| See Also..... | 1363 |
| Load SSRS Report Server reports | 1364 |
| Network credentials for SSRS | 1364 |
| Set data source credential to shared data sources..... | 1365 |
| Change data source connection string | 1365 |
| Render linked reports | 1365 |
| Load SharePoint Server reports | 1366 |
| Forms credential for SharePoint Server..... | 1366 |
| Set data source credential to shared data sources..... | 1366 |
| Load Bold Report Server reports | 1367 |
| Render RDLC report | 1369 |
| Bind data source in Web API controller..... | 1369 |
| Bind data source with razor view..... | 1372 |
| View report click | 1374 |
| Render subreport..... | 1374 |
| Change subreport path | 1377 |
| Set subreport parameter | 1377 |
| Modify subreport data source connection string | 1378 |
| Set subreport data source | 1378 |
| Load subreport stream | 1380 |

| | |
|----------------------------------------------------------|------|
| Report parameters..... | 1381 |
| Set a parameter with razor view..... | 1382 |
| Set parameters in Web API Controller..... | 1382 |
| Get report parameter | 1383 |
| Report interaction events | 1384 |
| Report loaded | 1384 |
| Report error | 1385 |
| Show error | 1386 |
| Drill through..... | 1386 |
| Hyperlink..... | 1387 |
| Handle post actions | 1387 |
| AjaxBeforeLoad..... | 1387 |
| Add custom header to Ajax request | 1387 |
| Pass custom data in Ajax request | 1388 |
| AjaxSuccess..... | 1390 |
| AjaxError | 1391 |
| Error logging in ASP.NET Core Report Viewer | 1391 |
| Print report | 1394 |
| View report in print mode | 1394 |
| Print in new page | 1395 |
| Set page orientation and paper size | 1395 |
| Set report margin..... | 1395 |
| Set page height and width | 1396 |
| Print report with images | 1397 |
| External styles in report printing | 1397 |
| Show print progress | 1398 |
| Remove empty spaces in printing..... | 1399 |
| Export report..... | 1399 |
| Export event handling..... | 1399 |
| Change Excel and Word export format..... | 1401 |
| Export data visualization items..... | 1401 |
| Export data visualization items in core environment..... | 1403 |
| Hide specific export type for report..... | 1404 |
| PDF export options | 1405 |
| Export with complex scripts..... | 1405 |

| | |
|----------------------------------------------------------------|------|
| PDF conformance | 1405 |
| Add custom PDF fonts..... | 1406 |
| Word export options..... | 1406 |
| Word document type..... | 1406 |
| Word document advance layout for merged cells | 1407 |
| Protecting Word document from editing | 1407 |
| Excel export options..... | 1408 |
| Excel document type..... | 1408 |
| Excel document advance layout for merged cells | 1408 |
| Protecting Excel document from editing | 1408 |
| CSV export options..... | 1409 |
| Password protect exported document..... | 1409 |
| Custom Actions | 1410 |
| Add email button | 1410 |
| Toolbar customization | 1412 |
| Hide parameter block and toolbar items..... | 1412 |
| Enable stop option in toolbar | 1413 |
| Hide toolbar | 1413 |
| Decide or hide the export option..... | 1414 |
| Add custom items to the export drop-down | 1414 |
| Add custom toolbar item | 1416 |
| Add custom item to exiting toolbar group..... | 1417 |
| Add new toolbar group | 1419 |
| Localization of Bold Reports ASP.NET Core Report Viewer | 1420 |
| Responsive layout rendering of ASP.NET Core Report Viewer..... | 1422 |
| Normal layout | 1422 |
| Responsive layout | 1422 |
| Limitations | 1422 |
| RDL specification..... | 1422 |
| Report layout | 1422 |
| Expressions..... | 1422 |
| SSRS..... | 1422 |
| Samples and Demos..... | 1422 |
| Locally installed reports | 1422 |
| Offline demos..... | 1423 |

| | |
|----------------------------------------------------------------------------------|------|
| Online demos | 1423 |
| GitHub demo samples..... | 1423 |
| Migrate Report Viewer application | 1423 |
| Client-side migration..... | 1423 |
| Scripts and CSS references..... | 1423 |
| Adding data visualization scripts..... | 1425 |
| Tag helper | 1426 |
| Control initialization..... | 1427 |
| Server-side migration..... | 1427 |
| Web API Controller | 1427 |
| NuGet Packages for ASP.NET Core..... | 1427 |
| Configure NuGet feed URL..... | 1428 |
| Online NuGet feed URL..... | 1428 |
| Offline NuGet feed URL..... | 1428 |
| Installing NuGet packages..... | 1428 |
| Install using NuGet Package Manager | 1428 |
| Install using Package Manager Console | 1429 |
| install specified package in default project | 1429 |
| install specified package in default project with specified package source | 1429 |
| install specified package in specified project..... | 1429 |
| install specified package in default project | 1429 |
| install specified package in default project with specified Package Source..... | 1429 |
| install specified package in specified project..... | 1429 |
| Upgrading NuGet packages | 1429 |
| Upgrading using NuGet Package Manager | 1429 |
| Upgrading using Package Manger Console..... | 1430 |
| Update specific NuGet package in default project | 1430 |
| Update all the packages in default project..... | 1430 |
| Update specified package in default project with specified package source..... | 1430 |
| Update specified package in specified project | 1430 |
| Update specified Bold Reporting NuGet package | 1430 |
| Update specified package in default project with specified Package Source..... | 1430 |
| Update specified package in specified project | 1430 |
| Upgrading using NuGet CLI | 1431 |
| update all NuGet packages from config file..... | 1431 |

| | |
|--------------------------------------------------------------------------|------|
| update all NuGet packages from specified Packages Source | 1431 |
| Update all NuGet packages from config file | 1431 |
| Update all NuGet packages from specified Packages Source | 1431 |
| How to Create RDL/RDLC Report..... | 1431 |
| Create a SSRS RDL report..... | 1432 |
| Bold Reports Report Designer | 1432 |
| Microsoft SQL Report Builder | 1432 |
| Visual Studio Report Server template..... | 1432 |
| Create a RDLC report using business object data source | 1432 |
| Prerequisites | 1432 |
| Create business object class | 1433 |
| Add an RDLC report..... | 1433 |
| Data source and table configuration wizard..... | 1433 |
| Render data visualization report items..... | 1434 |
| How to use the custom code with Report Viewer..... | 1435 |
| Create a custom code class file | 1436 |
| Create a assembly reference and add it to Report Viewer..... | 1436 |
| Display ssrs rdl report in Bold Reports ASP.NET Core Report Viewer | 1437 |
| Create ASP.NET Core application..... | 1437 |
| Enable Docker support..... | 1437 |
| install System.Drawing native dependencies | 1437 |
| List of dependency libraries | 1437 |
| Refer scripts and CSS..... | 1438 |
| Tag helper | 1439 |
| Configure Script Manager | 1440 |
| Initialize Report Viewer..... | 1440 |
| Add already created reports | 1440 |
| Configure Web API..... | 1440 |
| Add Web API Controller..... | 1441 |
| Enable cross-origin requests | 1443 |
| Set report path and service URL | 1444 |
| Preview the report | 1444 |
| See Also | 1445 |
| How to change the exporting document file name based on parameter | 1445 |
| How to change the connection string datasource dynamically..... | 1446 |

| | |
|-----------------------------------------------------------------------------------|------|
| How to disable the vertical scrollbar in parameter panel | 1447 |
| How to generate the RDL and RDLC reports programmatically with ASP.NET Core | 1447 |
| Initialize a report definition | 1448 |
| Create a report page header..... | 1448 |
| Create a report page footer..... | 1449 |
| Initialize a report body section | 1450 |
| Render the generated report in ReportViewer | 1453 |
| Frequently asked questions | 1453 |
| Is possible to change the culture of the date time parameter | 1454 |
| Is it possible to hide the parameters in Report Viewer | 1454 |
| Is it possible to hide the export options in Report Viewer | 1454 |
| Is it possible to load reports providing parameter values at runtime | 1454 |
| Bold Report Writer..... | 1454 |
| Export SSRS RDL Report in Bold Reports ASP.NET Core Report Writer | 1455 |
| Create ASP.NET Core application..... | 1455 |
| List of dependency libraries..... | 1455 |
| Server side Report Writer changes | 1456 |
| Client side changes..... | 1459 |
| Export RDLC Report..... | 1461 |
| Bind data source in Web API controller..... | 1461 |
| Export SSRS Report Server Report | 1465 |
| Network credentials for SSRS | 1466 |
| Set data source credential to shared data sources..... | 1466 |
| Render linked reports | 1468 |
| Export SharePoint Server Report | 1468 |
| Forms credential for SharePoint Server..... | 1469 |
| Set data source credential to shared data sources..... | 1469 |
| Export Subreport..... | 1471 |
| Export RDL Subreport | 1471 |
| Export RDLC subreport..... | 1474 |
| Export Parameter Report..... | 1475 |
| Set report parameters to export file..... | 1476 |
| Report Writer Events | 1479 |
| Subreport Processing..... | 1479 |
| Report Error | 1480 |

| | |
|------------------------------------------------------|------|
| Error logging in ASP.NET Core Report Writer | 1480 |
| Encrypt and Secure Documents..... | 1483 |
| Password Protected PDF document | 1483 |
| Password Protected Word document..... | 1484 |
| Password Protected Excel document | 1484 |
| Advance Layouts For Merged Cells | 1484 |
| Word document advance layout for merged cells | 1484 |
| Excel document advance layout for merged cells | 1485 |
| Change the Default File Format | 1485 |
| Change the Word document type | 1485 |
| Change the Excel document type | 1485 |
| Change the CSV document type | 1486 |
| PDF Settings | 1486 |
| Export with complex scripts..... | 1486 |
| PDF conformance..... | 1486 |
| Embedding Custom PDF fonts | 1486 |
| Password Protected PDF document | 1487 |
| Excel Settings | 1487 |
| Excel document type..... | 1487 |
| Excel document advance layout for merged cells | 1488 |
| Protecting Excel document from editing | 1488 |
| Change Excel export format..... | 1488 |
| Password Protected Excel document | 1488 |
| Word Settings | 1489 |
| Word document type..... | 1489 |
| Word document advance layout for merged cells | 1489 |
| Protecting Word document from editing | 1489 |
| Password Protected Word document..... | 1490 |
| CSV Settings | 1490 |
| Export Data Visualization in Core Environment..... | 1491 |
| Limitations | 1494 |
| Linux..... | 1494 |
| Custom code | 1494 |
| Data source | 1494 |
| PDF fonts..... | 1494 |

| | |
|-----------------------------------------------------------------------------------------|------|
| SSRS support | 1494 |
| Excel export..... | 1494 |
| How to section for Report Writer component | 1494 |
| How to use custom code with Report Writer | 1494 |
| Create a custom code class file..... | 1495 |
| Create a assembly reference and add it to Report Writer | 1495 |
| How to generate and export RDL and RDLC reports programmatically with ASP.NET Core..... | 1496 |
| Initialize a report definition | 1496 |
| Create a report page header..... | 1496 |
| Create a report page footer | 1497 |
| Initialize a report body section | 1498 |
| Export the generated report in ReportWriter | 1501 |
| Reporting tools for ASP.NET Core | 1501 |
| Key features | 1501 |
| Add Web Report Designer to an ASP.NET Core application | 1502 |
| List of dependency libraries..... | 1502 |
| Refer Scripts and CSS | 1503 |
| Configure Script Manager..... | 1505 |
| Tag helper | 1505 |
| Initialize Report Designer..... | 1505 |
| Add API Controller | 1506 |
| Configure Web API..... | 1506 |
| ReportDesignerHelper | 1506 |
| ReportHelper..... | 1506 |
| Set the service URL | 1510 |
| Run the Application..... | 1510 |
| Add Web Report Designer to an ASP.NET Core Razor Pages..... | 1510 |
| Prerequisites | 1510 |
| Create ASP.NET Core application..... | 1510 |
| List of dependency libraries | 1511 |
| Refer Scripts and CSS | 1511 |
| Tag helper | 1513 |
| Configure Script Manager | 1513 |
| Initialize Report Designer..... | 1513 |
| Add API Controller | 1513 |

| | |
|--------------------------------------------------|------|
| Configure Web API..... | 1514 |
| ReportDesignerHelper | 1514 |
| ReportHelper..... | 1514 |
| Web API with ASP.NET Core Web Application..... | 1518 |
| Set the service URL | 1518 |
| Run the Application..... | 1519 |
| Dependencies..... | 1519 |
| Scripts..... | 1519 |
| Styles | 1524 |
| Service side dependencies | 1526 |
| Internal dependencies | 1526 |
| External dependencies..... | 1526 |
| Localization | 1526 |
| Integrate the component with Report Server | 1529 |
| Designing Reports | 1531 |
| Connecting your data..... | 1531 |
| Transform your data | 1531 |
| Working with report parameters..... | 1532 |
| Embed images into the report..... | 1532 |
| Interacting with report design surface | 1532 |
| Report items | 1532 |
| Customize appearance..... | 1532 |
| Report design settings | 1532 |
| Shape report data | 1532 |
| Interactive features..... | 1532 |
| Working with Expressions..... | 1533 |
| Open report | 1533 |
| Save report..... | 1533 |
| Preview report | 1533 |
| Migrate Report Designer application..... | 1533 |
| Scripts..... | 1533 |
| Styles | 1535 |
| Assemblies | 1535 |
| Packages..... | 1535 |
| Tag helper | 1536 |

| | |
|----------------------------------------------------------------------------------|------|
| Configure script manager..... | 1536 |
| Control initialization..... | 1536 |
| NuGet Packages for ASP.NET Core..... | 1536 |
| Configure NuGet feed URL..... | 1536 |
| Online NuGet feed URL..... | 1536 |
| Offline NuGet feed URL..... | 1537 |
| Installing NuGet packages..... | 1537 |
| Install using NuGet Package Manager | 1537 |
| Install using Package Manager Console | 1537 |
| install specified package in default project | 1538 |
| install specified package in default project with specified package source | 1538 |
| install specified package in specified project..... | 1538 |
| install specified package in default project | 1538 |
| install specified package in default project with specified Package Source | 1538 |
| install specified package in specified project..... | 1538 |
| Upgrading NuGet packages | 1538 |
| Upgrading using NuGet Package Manager | 1538 |
| Upgrading using Package Manger Console..... | 1538 |
| Update specific NuGet package in default project..... | 1539 |
| Update all the packages in default project..... | 1539 |
| Update specified package in default project with specified package source..... | 1539 |
| Update specified package in specified project | 1539 |
| Update specified Bold Reporting NuGet package | 1539 |
| Update specified package in default project with specified Package Source..... | 1539 |
| Update specified package in specified project | 1539 |
| Upgrading using NuGet CLI | 1539 |
| update all NuGet packages from config file..... | 1540 |
| update all NuGet packages from specified Packages Source | 1540 |
| Update all NuGet packages from config file | 1540 |
| Update all NuGet packages from specified Packages Source | 1540 |
| Responsive layout rendering of ASP.NET Core Report Designer | 1540 |
| Normal layout | 1540 |
| Responsive layout | 1540 |
| Samples and Demos..... | 1540 |
| Offline demos..... | 1541 |

| | |
|----------------------------------------------------------------------------------------------|------|
| Online demos | 1541 |
| GitHub demo samples..... | 1541 |
| Supported Browsers..... | 1541 |
| How to queries for Bold Reports Report Designer | 1541 |
| How to open server report using report path at the time of opening the Report Designer | 1541 |
| How to add the data source and dataset for Report Designer from application | 1542 |
| Breaking Changes..... | 1546 |
| Breaking Changes..... | 1546 |
| Designer resource read and write API | 1546 |
| Parameters..... | 1546 |
| Return Type..... | 1547 |
| Code snippet | 1547 |
| Reporting tools for ASP.NET MVC..... | 1548 |
| How to best read this user guide | 1548 |
| Getting help | 1548 |
| Reporting tools for ASP.NET MVC..... | 1548 |
| How to best read this user guide | 1548 |
| Getting help | 1548 |
| System Requirements | 1549 |
| Supported Operating Systems | 1549 |
| Hardware Requirements..... | 1549 |
| Software Requirements | 1549 |
| Framework..... | 1549 |
| Browser Compatibility | 1549 |
| See Also..... | 1550 |
| Overview | 1550 |
| Display ssrs rdl report in Bold Reports ASP.NET MVC Report Viewer | 1550 |
| Create ASP.NET MVC 5 application..... | 1550 |
| Configure Report Viewer in an application..... | 1550 |
| ReportHelper..... | 1551 |
| Add Web API Controller..... | 1551 |
| Add routing information | 1552 |
| Set report path and service URL | 1553 |
| Preview the report..... | 1553 |
| See Also..... | 1553 |

| | |
|--------------------------------------------------------|------|
| Load SSRS Report Server reports | 1553 |
| Network credentials for SSRS | 1554 |
| Set data source credential to shared data sources..... | 1554 |
| Render linked reports | 1555 |
| Load SharePoint Server reports | 1555 |
| Forms credential for SharePoint Server..... | 1555 |
| Set data source credential to shared data sources..... | 1556 |
| Load Bold Report Server reports | 1556 |
| Render RDLC report | 1558 |
| Bind data source at client side..... | 1558 |
| Bind data source in Web API controller..... | 1560 |
| Load report as stream..... | 1561 |
| Render subreport..... | 1562 |
| Change subreport path | 1562 |
| Set subreport parameter | 1563 |
| Modify subreport data source connection string..... | 1563 |
| Set subreport data source | 1564 |
| Load subreport stream | 1566 |
| Report parameters..... | 1567 |
| Set parameter at client | 1567 |
| Set parameters in Web API Controller..... | 1567 |
| Get report parameter | 1568 |
| Report interaction events | 1569 |
| Report loaded | 1569 |
| Report error | 1570 |
| Show error | 1571 |
| Drill through..... | 1572 |
| Hyperlink..... | 1572 |
| Handle post actions | 1573 |
| AjaxBeforeLoad | 1573 |
| Add custom header to Ajax request | 1573 |
| Pass custom data in Ajax request | 1574 |
| AjaxSuccess | 1576 |
| AjaxError | 1576 |
| Error logging in ASP.NET MVC Report Viewer | 1577 |

| | |
|------------------------------------------------------------|------|
| Print report | 1580 |
| View report in print mode | 1580 |
| Print in new page | 1581 |
| Set page orientation and paper size | 1581 |
| Set report margin..... | 1581 |
| Set page height and width | 1582 |
| Print report with images | 1582 |
| External styles in report printing | 1583 |
| Show print progress..... | 1583 |
| Remove empty spaces in printing..... | 1584 |
| Export report..... | 1585 |
| Export event handling..... | 1585 |
| Export data visualization items..... | 1586 |
| Export data visualization items in azure environment | 1588 |
| Change Excel and Word export format..... | 1589 |
| Hide specific export type for report..... | 1589 |
| PDF export options | 1590 |
| Export with complex scripts..... | 1590 |
| PDF conformance..... | 1590 |
| Add custom PDF fonts..... | 1591 |
| Word export options..... | 1591 |
| Word document type..... | 1591 |
| Word document advance layout for merged cells | 1592 |
| Protecting Word document from editing | 1592 |
| Excel export options..... | 1593 |
| Excel document type..... | 1593 |
| Excel document advance layout for merged cells | 1593 |
| Protecting Excel document from editing | 1594 |
| CSV export options..... | 1594 |
| Password protect exported document | 1595 |
| Custom Actions | 1596 |
| Add email button | 1596 |
| Create custom email action | 1598 |
| Toolbar customization | 1600 |
| Hide parameter block and toolbar items..... | 1600 |

| | |
|---------------------------------------------------------------|------|
| Enable stop option in toolbar | 1600 |
| Hide toolbar | 1601 |
| Decide or hide the export option..... | 1601 |
| Add custom items to the export drop-down | 1602 |
| Add custom toolbar item | 1603 |
| Add custom item to exiting toolbar group..... | 1605 |
| Add new toolbar group | 1608 |
| Localization of Bold Reports ASP.NET MVC Report Viewer..... | 1609 |
| Responsive layout rendering of ASP.NET MVC Report Viewer..... | 1611 |
| Normal layout | 1611 |
| Responsive layout | 1611 |
| Limitations | 1611 |
| RDL specification..... | 1611 |
| Report layout | 1611 |
| Expressions..... | 1612 |
| SSRS..... | 1612 |
| Samples and Demos..... | 1612 |
| Locally installed reports | 1612 |
| Offline demos..... | 1612 |
| Online demos | 1612 |
| GitHub demo samples..... | 1612 |
| Migrate Report Viewer application | 1612 |
| Client-side migration..... | 1612 |
| Scripts and CSS references..... | 1613 |
| Adding data visualization scripts..... | 1614 |
| Control initialization..... | 1615 |
| Server-side migration..... | 1616 |
| Web API Controller | 1617 |
| Report export configuration | 1617 |
| NuGet Packages for ASP.NET MVC | 1619 |
| Configure NuGet feed URL..... | 1619 |
| Online NuGet feed URL..... | 1619 |
| Offline NuGet feed URL..... | 1619 |
| Installing NuGet packages..... | 1619 |
| Install using NuGet Package Manager | 1619 |

| | |
|----------------------------------------------------------------------------------|------|
| Install using Package Manager Console | 1620 |
| install specified package in default project | 1620 |
| install specified package in default project with specified package source | 1620 |
| install specified package in specified project..... | 1620 |
| install specified package in default project | 1620 |
| install specified package in default project with specified Package Source | 1620 |
| install specified package in specified project..... | 1620 |
| Upgrading NuGet packages | 1620 |
| Upgrading using NuGet Package Manager | 1620 |
| Upgrading using Package Manger Console..... | 1621 |
| Update specific NuGet package in default project | 1621 |
| Update all the packages in default project..... | 1621 |
| Update specified package in default project with specified package source..... | 1621 |
| Update specified package in specified project | 1621 |
| Update specified Bold Reporting NuGet package | 1621 |
| Update specified package in default project with specified Package Source..... | 1621 |
| Update specified package in specified project | 1621 |
| Upgrading using NuGet CLI | 1622 |
| update all NuGet packages from config file..... | 1622 |
| update all NuGet packages from specified Packages Source | 1622 |
| Update all NuGet packages from config file | 1622 |
| Update all NuGet packages from specified Packages Source | 1622 |
| Frequently asked questions | 1622 |
| Is possible to change the culture of the date time parameter | 1623 |
| Is it possible to hide the parameters in Report Viewer | 1623 |
| Is it possible to hide the export options in Report Viewer | 1623 |
| Is it possible to load reports providing parameter values at runtime | 1623 |
| How to Create RDL/RDLC Report..... | 1623 |
| Create a SSRS RDL report..... | 1624 |
| Bold Reports Report Designer | 1624 |
| Microsoft SQL Report Builder | 1624 |
| Visual Studio Report Server template..... | 1624 |
| Create a RDLC report using business object data source | 1624 |
| Prerequisites | 1624 |
| Create business object class | 1624 |

| | |
|-------------------------------------------------------------------------|------|
| Add an RDLC report..... | 1625 |
| Data source and table configuration wizard..... | 1625 |
| Adding data visualization scripts..... | 1626 |
| How to change the exporting document file name based on parameter | 1627 |
| How to change the connection string datasource dynamically..... | 1628 |
| How to disable the vertical scrollbar in parameter panel | 1629 |
| Bold Report Writer..... | 1630 |
| Export SSRS RDL Report in Bold Reports ASP.NET MVC Report Writer..... | 1630 |
| Create an ASP.NET MVC application..... | 1630 |
| List of dependency libraries | 1631 |
| Server side Report Writer changes | 1631 |
| Client side changes..... | 1634 |
| Reporting tools for ASP.NET MVC..... | 1636 |
| Key features | 1636 |
| Add Web Report Designer to an ASP.NET MVC application..... | 1637 |
| Prerequisites | 1637 |
| Create ASP.NET MVC 5 web application..... | 1637 |
| Add Assembly References..... | 1637 |
| Registering namespaces within Web.config | 1638 |
| Disable unobtrusive mode | 1638 |
| Refer Scripts and Styles..... | 1638 |
| Configure Script Manager | 1640 |
| Add Control in View page | 1640 |
| Add API controller..... | 1640 |
| Configure Web API..... | 1641 |
| ReportDesignerHelper | 1641 |
| ReportHelper..... | 1641 |
| Add routing information | 1646 |
| Set the service URL | 1647 |
| Run the Application..... | 1647 |
| Dependencies..... | 1647 |
| Scripts..... | 1647 |
| Styles | 1649 |
| Service side dependencies..... | 1650 |
| Internal dependencies | 1650 |

| | |
|--------------------------------------------------|------|
| External dependencies..... | 1650 |
| Localization | 1650 |
| Integrate the component with Report Server | 1653 |
| Designing Reports | 1655 |
| Connecting your data..... | 1655 |
| Transform your data | 1655 |
| Working with report parameters..... | 1655 |
| Embed images into the report..... | 1656 |
| Interacting with report design surface | 1656 |
| Report items | 1656 |
| Customize appearance..... | 1656 |
| Report design settings | 1656 |
| Shape report data | 1656 |
| Interactive features..... | 1656 |
| Working with Expressions..... | 1656 |
| Open report | 1656 |
| Save report..... | 1657 |
| Preview report | 1657 |
| Migrate Report Designer application..... | 1657 |
| Scripts..... | 1657 |
| Styles | 1658 |
| Assemblies | 1659 |
| Packages..... | 1659 |
| Register tag prefix in Web.config file..... | 1659 |
| Namespace changes | 1660 |
| Configure script manager..... | 1660 |
| Control initialization..... | 1660 |
| Report export configuration | 1660 |
| NuGet Packages for ASP.NET MVC | 1662 |
| Configure NuGet feed URL..... | 1662 |
| Online NuGet feed URL..... | 1662 |
| Offline NuGet feed URL..... | 1663 |
| Installing NuGet packages..... | 1663 |
| Install using NuGet Package Manager | 1663 |
| Install using Package Manager Console | 1663 |

| | |
|-----------------------------------------------------------------------------------|------|
| install specified package in default project | 1663 |
| install specified package in default project with specified package source | 1664 |
| install specified package in specified project..... | 1664 |
| install specified package in default project | 1664 |
| install specified package in default project with specified Package Source | 1664 |
| install specified package in specified project..... | 1664 |
| Upgrading NuGet packages | 1664 |
| Upgrading using NuGet Package Manager | 1664 |
| Upgrading using Package Manger Console..... | 1664 |
| Update specific NuGet package in default project | 1664 |
| Update all the packages in default project | 1665 |
| Update specified package in default project with specified package source..... | 1665 |
| Update specified package in specified project | 1665 |
| Update specified Bold Reporting NuGet package | 1665 |
| Update specified package in default project with specified Package Source..... | 1665 |
| Update specified package in specified project | 1665 |
| Upgrading using NuGet CLI | 1665 |
| update all NuGet packages from config file..... | 1666 |
| update all NuGet packages from specified Packages Source | 1666 |
| Update all NuGet packages from config file | 1666 |
| Update all NuGet packages from specified Packages Source | 1666 |
| Responsive layout rendering of ASP.NET MVC Report Designer | 1666 |
| Normal layout | 1666 |
| Responsive layout | 1666 |
| Samples and Demos..... | 1666 |
| Offline demos..... | 1666 |
| Online demos | 1667 |
| GitHub demo samples..... | 1667 |
| Supported Browsers..... | 1667 |
| How to queries for Bold Reports Report Designer | 1667 |
| How to open the RDL file at the time of opening the Report Designer..... | 1667 |
| How to add the data source and dataset for Report Designer from application | 1668 |
| Breaking Changes..... | 1672 |
| Breaking Changes..... | 1672 |
| Designer resource read and write API | 1672 |

| | |
|-------------------------------------------------------------------------------|------|
| Parameters..... | 1672 |
| Return Type..... | 1673 |
| Code snippet | 1673 |
| Reporting tools for ASP.NET Web Forms..... | 1674 |
| How to best read this user guide | 1674 |
| Getting help | 1674 |
| Reporting tools for ASP.NET Web Forms..... | 1674 |
| How to best read this user guide | 1674 |
| Getting help | 1675 |
| System Requirements..... | 1675 |
| Supported Operating Systems | 1675 |
| Hardware Requirements..... | 1675 |
| Software Requirements | 1675 |
| Framework..... | 1675 |
| Browser Compatibility | 1675 |
| See Also..... | 1676 |
| Overview | 1676 |
| Display ssrs rdl report in Bold Reports ASP.NET Web Forms Report Viewer | 1676 |
| Create ASP.NET Web Forms application..... | 1676 |
| Configure Report Viewer in an application..... | 1676 |
| ReportHelper..... | 1677 |
| Add Web API Controller..... | 1677 |
| Add routing information | 1678 |
| Set report path and service URL | 1679 |
| Preview the report..... | 1680 |
| See Also..... | 1680 |
| Load SSRS Report Server reports | 1680 |
| Network credentials for SSRS | 1680 |
| Set data source credential to shared data sources..... | 1681 |
| Render linked reports | 1681 |
| Load SharePoint Server reports | 1681 |
| Forms credential for SharePoint Server..... | 1682 |
| Set data source credential to shared data sources..... | 1682 |
| Load Bold Report Server reports | 1683 |
| Render RDLC report | 1685 |

| | |
|--------------------------------------------------------|------|
| Bind data source at client | 1685 |
| Bind data source in Web API controller..... | 1686 |
| Render subreport..... | 1688 |
| Change subreport path | 1688 |
| Set subreport parameter | 1689 |
| Modify subreport data source connection string..... | 1689 |
| Set subreport data source | 1690 |
| Load subreport stream | 1692 |
| Report parameters..... | 1693 |
| Set parameter at client | 1693 |
| Set parameters in Web API Controller..... | 1694 |
| Get report parameter | 1694 |
| Report interaction events | 1695 |
| Report loaded | 1696 |
| Report error | 1697 |
| Show error | 1698 |
| Drill through..... | 1698 |
| Hyperlink..... | 1699 |
| Handle post actions | 1699 |
| AjaxBeforeLoad | 1700 |
| Add custom header to Ajax request | 1700 |
| Pass custom data in Ajax request | 1701 |
| AjaxSuccess | 1703 |
| AjaxError | 1703 |
| Error logging in ASP.NET Web Forms Report Viewer | 1704 |
| Print report | 1707 |
| View report in print mode | 1707 |
| Print in new page | 1708 |
| Set page orientation and paper size | 1708 |
| Set report margin..... | 1709 |
| Set page height and width..... | 1710 |
| Print report with images | 1710 |
| External styles in report printing | 1711 |
| Show print progress | 1712 |
| Remove empty spaces in printing..... | 1713 |

| | |
|--------------------------------------------------------------------|------|
| Export report..... | 1713 |
| Export event handling..... | 1713 |
| Export data visualization items..... | 1715 |
| Export data visualization items in azure environment | 1716 |
| Change Excel and Word export format..... | 1718 |
| Hide specific export type for report..... | 1718 |
| PDF export options | 1719 |
| Export with complex scripts..... | 1719 |
| PDF conformance..... | 1719 |
| Add custom PDF fonts..... | 1720 |
| Word export options..... | 1721 |
| Word document type..... | 1721 |
| Word document advance layout for merged cells | 1721 |
| Protecting Word document from editing | 1722 |
| Excel export options..... | 1722 |
| Excel document type..... | 1722 |
| Excel document advance layout for merged cells | 1723 |
| Protecting Excel document from editing | 1723 |
| CSV export options..... | 1723 |
| Password protect exported document..... | 1724 |
| Toolbar customization | 1725 |
| Hide parameter block and toolbar items..... | 1725 |
| Enable stop option in toolbar | 1726 |
| Hide toolbar | 1726 |
| Decide or hide the export option..... | 1727 |
| Add custom items to the export drop-down | 1728 |
| Add custom toolbar item | 1729 |
| Add custom item to exiting toolbar group..... | 1730 |
| Add new toolbar group | 1732 |
| Custom Actions | 1732 |
| Add email button | 1733 |
| Create custom email action | 1735 |
| Localization of Bold Reports ASP.NET Web Forms Report Viewer | 1736 |
| Responsive layout rendering of ASP.NET Report Viewer | 1738 |
| Normal layout | 1738 |

| | |
|----------------------------------------------------------------------------------|------|
| Responsive layout | 1738 |
| Limitations | 1738 |
| RDL specification..... | 1738 |
| Report layout | 1738 |
| Expressions..... | 1738 |
| SSRS..... | 1739 |
| Samples and Demos..... | 1739 |
| Locally installed reports | 1739 |
| Offline demos..... | 1739 |
| Online demos | 1739 |
| GitHub demo samples..... | 1739 |
| Migrate Report Viewer application | 1739 |
| Client-side migration..... | 1739 |
| Scripts and CSS references..... | 1740 |
| Adding data visualization scripts..... | 1740 |
| Control initialization..... | 1741 |
| Server-side migration..... | 1742 |
| Web API Controller | 1743 |
| Report export configuration | 1743 |
| NuGet Packages for ASP.NET Web Forms..... | 1745 |
| Configure NuGet feed URL..... | 1745 |
| Online NuGet feed URL | 1745 |
| Offline NuGet feed URL..... | 1745 |
| Installing NuGet packages..... | 1746 |
| Install using NuGet Package Manager | 1746 |
| Install using Package Manager Console | 1746 |
| install specified package in default project | 1746 |
| install specified package in default project with specified package source | 1746 |
| install specified package in specified project..... | 1746 |
| install specified package in default project | 1746 |
| install specified package in default project with specified Package Source..... | 1746 |
| install specified package in specified project..... | 1746 |
| Upgrading NuGet packages | 1747 |
| Upgrading using NuGet Package Manager | 1747 |
| Upgrading using Package Manger Console..... | 1747 |

| | |
|--------------------------------------------------------------------------------|------|
| Update specific NuGet package in default project | 1747 |
| Update all the packages in default project..... | 1747 |
| Update specified package in default project with specified package source..... | 1747 |
| Update specified package in specified project | 1747 |
| Update specified Bold Reporting NuGet package | 1747 |
| Update specified package in default project with specified Package Source..... | 1748 |
| Update specified package in specified project | 1748 |
| Upgrading using NuGet CLI | 1748 |
| update all NuGet packages from config file..... | 1748 |
| update all NuGet packages from specified Packages Source | 1748 |
| Update all NuGet packages from config file | 1748 |
| Update all NuGet packages from specified Packages Source | 1748 |
| Frequently asked questions | 1749 |
| Is possible to change the culture of the date time parameter | 1749 |
| Is it possible to hide the parameters in Report Viewer | 1749 |
| Is it possible to hide the export options in Report Viewer | 1749 |
| Is it possible to load reports providing parameter values at runtime | 1749 |
| How to Create RDL/RDLC Report..... | 1750 |
| Create a SSRS RDL report..... | 1750 |
| Bold Reports Report Designer | 1750 |
| Microsoft SQL Report Builder | 1750 |
| Visual Studio Report Server template..... | 1750 |
| Create a RDLC report using business object data source | 1750 |
| Prerequisites | 1750 |
| Create business object class | 1751 |
| Add an RDLC report..... | 1751 |
| Data source and table configuration wizard..... | 1751 |
| Adding data visualization scripts..... | 1752 |
| How to change the exporting document file name based on parameter | 1753 |
| How to change the connection string datasource dynamically..... | 1754 |
| How to disable the vertical scrollbar in parameter panel | 1756 |
| Bold Report Writer..... | 1756 |
| Export SSRS RDL Report in Bold Reports ASP.NET Report Writer | 1757 |
| Create an ASP.NET application | 1757 |
| List of dependency libraries | 1757 |

| | |
|------------------------------------------------------------------|------|
| Server side Report Writer changes | 1757 |
| Client side changes..... | 1759 |
| Reporting tools for ASP.NET Web Forms..... | 1761 |
| Key features | 1761 |
| Add Web Report Designer to an ASP.NET WebForms application | 1761 |
| Create ASP.NET Web Forms application..... | 1761 |
| Add Assembly References..... | 1762 |
| Add routing information | 1766 |
| Set the service URL | 1767 |
| Run the Application..... | 1768 |
| Dependencies..... | 1768 |
| Scripts..... | 1769 |
| Styles | 1771 |
| Service side dependencies | 1771 |
| Internal dependencies | 1771 |
| External dependencies..... | 1772 |
| Localization | 1772 |
| Integrate the component with Report Server | 1774 |
| Designing Reports | 1775 |
| Connecting your data..... | 1776 |
| Transform your data | 1776 |
| Working with report parameters..... | 1776 |
| Embed images into the report..... | 1776 |
| Interacting with report design surface | 1776 |
| Report items | 1776 |
| Customize appearance..... | 1776 |
| Report design settings | 1777 |
| Shape report data | 1777 |
| Interactive features..... | 1777 |
| Working with Expressions..... | 1777 |
| Open report | 1777 |
| Save report..... | 1777 |
| Preview report | 1777 |
| Migrate Report Designer application..... | 1777 |
| Scripts..... | 1777 |

| | |
|----------------------------------------------------------------------------------|------|
| Styles | 1779 |
| Assemblies | 1779 |
| Packages..... | 1780 |
| Register tag prefix in Web.config file..... | 1780 |
| Namespace changes | 1780 |
| Control initialization..... | 1781 |
| Report export configuration | 1781 |
| NuGet Packages for ASP.NET Web Forms..... | 1782 |
| Configure NuGet feed URL..... | 1782 |
| Online NuGet feed URL..... | 1782 |
| Offline NuGet feed URL..... | 1783 |
| Installing NuGet packages..... | 1783 |
| Install using NuGet Package Manager | 1783 |
| Install using Package Manager Console | 1784 |
| install specified package in default project | 1784 |
| install specified package in default project with specified package source | 1784 |
| install specified package in specified project..... | 1784 |
| install specified package in default project | 1784 |
| install specified package in default project with specified Package Source | 1784 |
| install specified package in specified project..... | 1784 |
| Upgrading NuGet packages | 1784 |
| Upgrading using NuGet Package Manager | 1784 |
| Upgrading using Package Manger Console..... | 1785 |
| Update specific NuGet package in default project | 1785 |
| Update all the packages in default project | 1785 |
| Update specified package in default project with specified package source..... | 1785 |
| Update specified package in specified project | 1785 |
| Update specified Bold Reporting NuGet package | 1785 |
| Update specified package in default project with specified Package Source..... | 1785 |
| Update specified package in specified project | 1785 |
| Upgrading using NuGet CLI | 1785 |
| update all NuGet packages from config file..... | 1786 |
| update all NuGet packages from specified Packages Source | 1786 |
| Update all NuGet packages from config file | 1786 |
| Update all NuGet packages from specified Packages Source | 1786 |

| | |
|---------------------------------------------------------------------------------------------|------|
| Responsive layout rendering of ASP.NET Report Designer | 1786 |
| Normal layout | 1786 |
| Responsive layout..... | 1786 |
| Samples and Demos..... | 1787 |
| Offline demos..... | 1787 |
| Online demos | 1787 |
| GitHub demo samples..... | 1787 |
| Supported Browsers..... | 1787 |
| How to queries for Bold Reports Report Designer | 1787 |
| How to open server report using report path at the time of opening the Report Designer..... | 1787 |
| How to add the data source and dataset for Report Designer from application | 1788 |
| Breaking Changes..... | 1792 |
| Breaking Changes..... | 1792 |
| Designer resource read and write API | 1792 |
| Parameters..... | 1792 |
| Return Type..... | 1793 |
| Code snippet | 1794 |
| Reporting tools for WPF..... | 1794 |
| How to best read this user guide | 1794 |
| Getting help | 1794 |
| Reporting tools for WPF..... | 1794 |
| How to best read this user guide | 1794 |
| Getting help | 1795 |
| System Requirements..... | 1795 |
| Supported Operating Systems | 1795 |
| Hardware Requirements..... | 1795 |
| Software Requirements | 1795 |
| Framework | 1795 |
| See Also | 1795 |
| Overview | 1795 |
| Display ssrs rdl report in Bold Reports WPF Report Viewer | 1796 |
| Create the application project | 1796 |
| Configure Report Viewer in an application..... | 1796 |
| Initialize Report Viewer..... | 1796 |
| Add already created reports | 1797 |

| | |
|----------------------------------------------------------|------|
| Set report path..... | 1797 |
| Preview the report..... | 1798 |
| See Also..... | 1798 |
| Load SSRS rdl reports | 1798 |
| Set data source credential for shared data sources | 1799 |
| Render linked reports | 1799 |
| Load SharePoint integrated reports | 1800 |
| Set data source credential for shared data sources | 1800 |
| Load Bold report server reports..... | 1801 |
| Render RDLC report | 1802 |
| Load report as stream..... | 1804 |
| Render subreport..... | 1805 |
| Load subreport stream | 1806 |
| Report parameters..... | 1808 |
| Set report parameter | 1808 |
| Get report parameter | 1809 |
| Report interaction events | 1809 |
| Report loaded | 1810 |
| Report error | 1810 |
| Drill through | 1810 |
| Hyperlink..... | 1811 |
| Error logging in WPF Report Viewer | 1812 |
| Print report | 1814 |
| View report in print mode | 1815 |
| Remove empty spaces in printing..... | 1815 |
| Export report..... | 1815 |
| PDF export options | 1815 |
| Export with complex scripts..... | 1815 |
| PDF Conformance | 1815 |
| Word export options..... | 1816 |
| Word document type..... | 1816 |
| Word document advance layout for merged cells | 1816 |
| Protecting Word document from editing | 1816 |
| Excel export options..... | 1817 |
| Excel document type..... | 1817 |

| | |
|----------------------------------------------------------------------------------|------|
| Excel document advance layout for merged cells | 1817 |
| Protecting Excel document from editing | 1817 |
| PowerPoint export options..... | 1817 |
| CSV export options..... | 1818 |
| HTML export options | 1818 |
| Password protect exported document..... | 1818 |
| Localization of Bold Report Viewer..... | 1819 |
| Add the Resource file for the different cultures..... | 1819 |
| Assign the values to each culture using key | 1820 |
| Assign a Current UI Culture to the application | 1820 |
| Set Report Viewer properties | 1820 |
| Limitations | 1821 |
| RDL specification..... | 1821 |
| Report layout | 1821 |
| Expressions..... | 1821 |
| SSRS..... | 1821 |
| Samples and Demos..... | 1821 |
| Locally installed reports | 1821 |
| Offline demos..... | 1821 |
| Migrate Report Viewer application | 1821 |
| Client-side migration..... | 1822 |
| Control initialization..... | 1822 |
| NuGet Packages for WPF | 1822 |
| Configure NuGet feed URL..... | 1822 |
| Online NuGet feed URL..... | 1822 |
| Installing NuGet packages..... | 1823 |
| Install using NuGet Package Manager | 1823 |
| Install using Package Manager Console | 1823 |
| install specified package in default project | 1823 |
| install specified package in default project with specified package source | 1823 |
| install specified package in specified project..... | 1823 |
| install specified package in default project | 1823 |
| install specified package in default project with specified Package Source | 1824 |
| install specified package in specified project..... | 1824 |
| Upgrading NuGet packages | 1824 |

| | |
|-----------------------------------------------------------------------------------|------|
| Upgrading using NuGet Package Manager | 1824 |
| Upgrading using Package Manger Console..... | 1824 |
| Update specific NuGet package in default project | 1824 |
| Update all the packages in default project..... | 1824 |
| Update specified package in default project with specified package source..... | 1824 |
| Update specified package in specified project | 1824 |
| Update specified Bold Reporting NuGet package | 1825 |
| Update specified package in default project with specified Package Source..... | 1825 |
| Update specified package in specified project | 1825 |
| Upgrading using NuGet CLI | 1825 |
| update all NuGet packages from config file..... | 1825 |
| update all NuGet packages from specified Packages Source | 1825 |
| Update all NuGet packages from config file | 1825 |
| Update all NuGet packages from specified Packages Source | 1826 |
| How to Create RDL/RDLC Report..... | 1826 |
| Create a SSRS RDL report | 1826 |
| Bold Reports Report Designer | 1826 |
| Microsoft SQL Report Builder | 1826 |
| Visual Studio Report Server template..... | 1826 |
| Create a RDLC report using business object data source | 1826 |
| Prerequisites | 1827 |
| Create business object class | 1827 |
| Add an RDLC report..... | 1827 |
| Data source and table configuration wizard..... | 1828 |
| Bold Report Writer..... | 1828 |
| Export SSRS RDL Report in Bold Reports WPF Report Writer..... | 1829 |
| Create WPF application..... | 1829 |
| List of dependency libraries..... | 1829 |
| Add already created reports | 1829 |
| Initialize Report Writer | 1830 |
| Set report path..... | 1831 |
| Frequently asked questions | 1833 |
| How to print the report silently without preview the report in Report Viewer..... | 1833 |
| Reporting tools for UWP | 1834 |
| How to best read this user guide | 1835 |

| | |
|-----------------------------------------------------------------|------|
| Getting help | 1835 |
| Reporting tools for UWP | 1835 |
| How to best read this user guide | 1835 |
| Getting help | 1835 |
| System Requirements | 1835 |
| Supported Operating Systems | 1835 |
| Hardware Requirements..... | 1835 |
| Software Requirements | 1836 |
| Framework..... | 1836 |
| See Also..... | 1836 |
| Overview | 1836 |
| Display ssrs rdl report in Bold Reports UWP Report Viewer | 1836 |
| Create the application project | 1836 |
| Configure Report Viewer in an application..... | 1836 |
| Initialize Report Viewer..... | 1837 |
| Create Web API service..... | 1837 |
| Adding already created report..... | 1838 |
| Set report path and service URL | 1838 |
| Preview the report..... | 1839 |
| See Also..... | 1839 |
| Load SSRS Report Server reports | 1839 |
| Set data source credential for shared data sources | 1839 |
| Render linked reports | 1840 |
| Load SharePoint Server reports | 1840 |
| Set data source credential for shared data sources | 1841 |
| Load Bold Report Server reports | 1841 |
| Render RDLC report | 1843 |
| Render subreport..... | 1845 |
| Load subreport stream | 1845 |
| Report parameters..... | 1847 |
| Set report parameter | 1848 |
| Set report parameter in Web API | 1848 |
| Get report parameter | 1849 |
| Report interaction events | 1850 |
| Report loaded | 1851 |

| | |
|------------------------------------------------------|------|
| Report error | 1851 |
| Drill through | 1852 |
| Error logging in UWP Report Viewer | 1852 |
| Print report | 1855 |
| Remove empty spaces in printing..... | 1855 |
| Export report..... | 1855 |
| PDF export options | 1855 |
| Export with complex scripts..... | 1855 |
| PDF Conformance | 1856 |
| Add custom PDF fonts..... | 1856 |
| Word export options..... | 1856 |
| Word document type..... | 1856 |
| Word document advance layout for merged cells | 1857 |
| Protecting Word document from editing | 1857 |
| Excel export options..... | 1858 |
| Excel document type..... | 1858 |
| Excel document advance layout for merged cells | 1858 |
| Protecting Excel document from editing | 1858 |
| PowerPoint export options..... | 1858 |
| CSV export options..... | 1858 |
| HTML export options | 1859 |
| Password protect exported document..... | 1859 |
| Handle post actions | 1860 |
| RenderingBegin..... | 1860 |
| RenderingCompleted | 1861 |
| ReportServiceRequestBegin..... | 1861 |
| EncryptCredentials..... | 1861 |
| Localization of Bold Report Viewer..... | 1862 |
| Add Resources file for the different cultures..... | 1862 |
| Assign the value to each culture using key | 1862 |
| Assign a Current UI Culture to the application | 1863 |
| Set Report Viewer properties | 1863 |
| Limitations | 1863 |
| RDL specification..... | 1863 |
| Report layout | 1863 |

| | |
|----------------------------------------------------------------------------------|------|
| Expressions..... | 1864 |
| SSRS..... | 1864 |
| Samples and Demos..... | 1864 |
| Locally installed reports | 1864 |
| Offline demos..... | 1864 |
| Migrate Report Viewer application | 1864 |
| Client-side migration..... | 1864 |
| Control initialization..... | 1864 |
| Server-side migration..... | 1865 |
| Web API Controller | 1865 |
| Report export configuration | 1865 |
| NuGet Packages for UWP..... | 1867 |
| Configure NuGet feed URL..... | 1867 |
| Online NuGet feed URL | 1867 |
| Installing NuGet packages..... | 1867 |
| Install using NuGet Package Manager | 1867 |
| Install using Package Manager Console | 1868 |
| install specified package in default project | 1868 |
| install specified package in default project with specified package source | 1868 |
| install specified package in specified project..... | 1868 |
| install specified package in default project | 1868 |
| install specified package in default project with specified Package Source | 1868 |
| install specified package in specified project..... | 1868 |
| Upgrading NuGet packages | 1868 |
| Upgrading using NuGet Package Manager | 1868 |
| Upgrading using Package Manger Console..... | 1869 |
| Update specific NuGet package in default project..... | 1869 |
| Update all the packages in default project..... | 1869 |
| Update specified package in default project with specified package source..... | 1869 |
| Update specified package in specified project | 1869 |
| Update specified Bold Reporting NuGet package | 1869 |
| Update specified package in default project with specified Package Source..... | 1869 |
| Update specified package in specified project | 1869 |
| Upgrading using NuGet CLI | 1870 |
| update all NuGet packages from config file..... | 1870 |

| | |
|----------------------------------------------------------------|------|
| update all NuGet packages from specified Packages Source | 1870 |
| Update all NuGet packages from config file | 1870 |
| Update all NuGet packages from specified Packages Source | 1870 |
| Frequently asked questions | 1870 |
| UWP Report Viewer Reporting Service..... | 1871 |
| Create ASP.NET Web API service | 1871 |
| Configure Report Viewer Web API..... | 1871 |
| ReportHelper..... | 1872 |
| Add Web API Controller..... | 1872 |
| Add routing information | 1873 |
| Enable Cross-Origin Requests | 1874 |
| Create ASP.NET Core Web API Service | 1876 |
| List of dependency Libraries | 1876 |
| Configure Web API..... | 1876 |
| Add Web API Controller..... | 1877 |
| Enable Cross-Origin requests | 1880 |
| How to Create RDL/RDLC Report..... | 1880 |
| Create a SSRS RDL report..... | 1881 |
| Bold Reports Report Designer | 1881 |
| Microsoft SQL Report Builder | 1881 |
| Visual Studio Report Server template..... | 1881 |
| Create a RDLC report using business object data source | 1881 |
| Prerequisites | 1881 |
| Create business object class | 1881 |
| Add an RDLC report..... | 1882 |
| Data source and table configuration wizard..... | 1882 |
| Bold Report Writer..... | 1883 |
| Export SSRS RDL Report in Bold Reports UWP Report Writer | 1884 |
| Create UWP application..... | 1884 |
| Configure Report Writer in an application..... | 1884 |
| Adding already created report..... | 1884 |
| Initialize Report Writer | 1885 |
| Set Report Path..... | 1886 |
| Embedded reporting tools licensing overview | 1890 |
| How to generate Bold Reports license token | 1890 |

| | |
|-----------------------------------------------------------------------------------------------------------------------------------------------|------|
| How to register the Bold Reports license tokens | 1891 |
| ASP.NET | 1891 |
| ASP.NET MVC | 1891 |
| ASP.NET Core | 1891 |
| WPF | 1892 |
| UWP | 1892 |
| Embedded reporting tools licensing overview | 1893 |
| How to generate Bold Reports license token | 1893 |
| How to register the Bold Reports license token | 1893 |
| ASP.NET | 1893 |
| ASP.NET MVC | 1893 |
| ASP.NET Core | 1894 |
| WPF | 1894 |
| UWP | 1895 |
| Embedded reporting tools license token errors | 1895 |
| License token not registered | 1895 |
| Invalid token | 1895 |
| License Expired | 1895 |
| Platform Mismatch | 1896 |
| Internet Connection Error | 1896 |
| Network Error | 1896 |
| Could not load Bold.Licensing.dll assembly version | 1896 |
| FAQ section for Bold Licensing..... | 1896 |
| How to upgrade from Trial version after purchasing a license..... | 1897 |
| Where can I get a license token..... | 1897 |
| Does Bold Reports Embedded Reporting Tools or Viewer SDK require additional licensing when deploying an application to Azure App service..... | 1897 |
| See also | 1897 |
| Can Syncfusion licenses be used with Bold Reports | 1898 |
| See also | 1898 |
| Can the Syncfusion community license be used with Bold Reports | 1898 |
| See also | 1898 |
| Can the Report Viewer component be accessed from Bold Reports using Syncfusion community license | 1898 |
| See also | 1898 |

| | |
|----------------------------------------------------------------------------------------------------------|------|
| Can the Report Designer component be accessed from Bold Reports using Syncfusion community license | 1898 |
| See also | 1898 |
| Can the Report Designer component be used from Bold Reports using Syncfusion license | 1899 |
| See also | 1899 |
| Is Report Designer included with Report Viewer SDK to create Reports | 1899 |
| What are the differences in Bold Reports licensing models..... | 1899 |
| See also | 1900 |
| Embedded reporting tools licensing version 1.2.x..... | 1900 |
| Embedded reporting tools licensing overview | 1900 |
| How to generate Bold Reports license key | 1900 |
| How to register the Bold Reports license key..... | 1901 |
| ASP.NET | 1901 |
| ASP.NET MVC | 1901 |
| ASP.NET Core | 1901 |
| WPF | 1902 |
| UWP | 1902 |
| Embedded reporting tools license key errors..... | 1903 |
| License key not registered | 1903 |
| Invalid key | 1903 |
| Trial Expired | 1903 |
| License Expired | 1903 |
| Platform Mismatch | 1904 |
| Version Mismatch | 1904 |
| Could not load Bold.Licensing.dll assembly version | 1904 |

Overview

The Bold Reports is an online solution for analyzing data in web using the interactive reports with key metrics and tracking your business easily.

It is an end-to-end solution for creating, managing, and sharing interactive business report that includes a powerful report designer for composing easily.

It does not required any installation and maintenance.

Key features

Wide variety of date sources: All the commonly used datasource such as Microsoft SQL Server, PostgreSQL and all the RESTful web services are supported.

Rich selection of data visualizations: Bold Report Cloud include charts, grids and several powerful filtering possibilities using data picker and combo boxes.

Business user friendly: Drag-friendly report designer allows business user to compose and publish report without any help from IT

Report management: Reports can be efficiently organized under the categories. Permission to view the reports can be given to specific users or groups.

Cost effective licensing: The licensing cost is effective for both small and large teams. Check the [pricing plans](#)

User management: Users can be easily organized into groups to accurately map the structure of the small and large organizations.

Scheduling: Reports can be generated and emailed according to a schedule. The scheduling functionality is very flexible.

Flexible permissions: A flexible permission scheme controls the access to read, write, and delete reports

Custom branding: The Report Server has built-in customization capabilities that allows you to add your organization's name, logo, welcome note, etc.

Create a support incident

Still, if you are not able to find the information that you are looking for Report Server in the self-help resources mentioned above, please [contact us](#) by creating a support ticket.

Overview

The Bold Reports is an online solution for analyzing data in web using the interactive reports with key metrics and tracking your business easily.

It is an end-to-end solution for creating, managing, and sharing interactive business report that includes a powerful report designer for composing easily.

We can easily create report with no software to installation and no IT required. All you need is a computer with a modern browser with internet connection.

Key features

Wide variety of date sources: All the commonly used datasource such as Microsoft SQL Server, PostgreSQL and all the RESTful web services are supported.

Rich selection of data visualizations: Bold Report Cloud include charts, grids and several powerful filtering possibilities using data picker and combo boxes.

Business user friendly: Drag-friendly report designer allows business user to compose and publish report without any help from IT

Report management: Reports can be efficiently organized under the categories. Permission to view the reports can be given to specific users or groups.

User management: Users can be easily organized into groups to accurately map the structure of the small and large organizations.

Scheduling: Reports can be generated and emailed according to a schedule. The scheduling functionality is very flexible.

Flexible permissions: A flexible permission scheme controls the access to read, write, and delete reports

Custom branding: The Report Server has built-in customization capabilities that allows you to add your organization's name, logo, welcome note, etc.

[Create a support incident](#)

Still, if you are not able to find the information that you are looking for Report Server in the self-help resources mentioned above, please [contact us](#) by creating a support ticket.

Overview

The Bold Reports Cloud is an end-to-end solution for creating, managing and sharing interactive business reports without any installation and maintenance.

[Registration](#)

Please follow the below steps to deploy the tenants in Bold Reports Cloud.

1. Go to BoldReports [link](#) to register and deploy the tenants in Bold Reports Cloud.
2. Select Cloud from Product Category.

![Bold Reports Cloud Plans](/static/assets/cloud/images/getting-started/trail-page.png)

3. Choose your plan based on your requirements.

![Bold Reports Trail Plans Page](/static/assets/cloud/images/getting-started/trail-plans.png)

4. Create your Bold Reports Cloud portal by creating new account or sign in with existing account.

![Bold Reports Portal Registration](/static/assets/cloud/images/getting-started/portal-registration.png)

5. Once portal have been created, your tenant will be deploy.

![Bold Report Deployment](/static/assets/cloud/images/getting-started/deployment.png)

6. Then it redirect to your Report Home.

![Bold Report Home](/static/assets/cloud/images/getting-started/bold-report-home.png)

Manage Users and Groups

Manage User

- This section explains on how to add, edit, activate, deactivate, delete users and also on how to manage the permissions and assign users to groups in the Bold Reports Server.

[Manage User](#)

Import Users

- You can add user to the report server by using the Azure Active Directory, Import user from CSV and also add user from Database.

[Import Azure Active Directory Users](#)

[Import User From CSV](#)

[Import User From Database](#)

Manage Group

- This section explains on how to add, edit, delete groups and also on how to assign users and manage permissions to groups in the Bold Reports Server.

[Manage Group](#)

Import Groups

- You can add group to the report server by using the Azure Active Directory.

[Import Azure Active Directory Group](#)

Manage Permission

- Permissions can be managed only by the users belonging to the System administrator. Permissions can be directly added to both users and groups and can be classified in the Access Mode – Entity – Scope structure.

[Manage Permissions](#)

Manage Report

Handling report in the bold report cloud is made easier you can easily create, edit, upload, update, share, clone, copy, move and delete report in the Bold Report Cloud.

Create Report

User can Create a report when user have the all report create permission

[Create Report](#)

[Edit Report](#)

User can Edit the selected report using report designer

[Edit Report](#)[Upload Report](#)

User can Upload a RDL report from your local server

[Upload Report](#)[Update Report](#)

You can Update a selected report Name, Dataset and Datasource

[Update report](#)[Move,Clone Report](#)

You can easily move,clone report from one category to another category

[Move Report](#)[Clone Report](#)[Copy Report](#)

You can copy report from one to another category

![Copy Report](/static/assets/cloud/images/manage-report/copy-report.png)

[Share Report](#)

You can share the report and manage the user permission

[Share Report](#)[Delete Report](#)

By selecting the Report the user can delete report from the server

[Delete Report](#)[Public Report](#)

By selecting report as public every user in the tenant can view the report

[Public Report](#)[Manage Categories](#)

User can add new category, modify the existing category and delete the category

[Manage Categories](#)

Designing Reports

The Bold Report Designer provides an end-user documentation that describes how to interact with the Report Designer's UI and design an interactive reports. Refer the following user guide sections to get start with Bold Report Designer.

Connecting your data

A report must have a data source and dataset to include data. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

[Manage data source](#)[Manage data set](#)

Transform your data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)[Formatting columns](#)[Query filter](#)[Data set parameter](#)[Query parameter](#)[Configure expression columns](#)

Working with report parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it.

[Add report parameter](#)[Edit report parameter](#)[Delete report parameter](#)[Cascading parameter](#)[Multi-value parameter](#)

Embed images into the report

Supports to embed local or database images into the report and image data is stored within the report definition. The embedded images in a report are listed in the **Image Manager panel**.

[Embed images into the report](#)

Interacting with report design surface

WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

[Interacting with report design surface](#)

Report items

Offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

[Configure report items](#)

Customize appearance

The Properties panel shows all the properties of the selected section, report items or the report itself to customize the appearance of the report.

[Customize appearance](#)

Report design settings

[Configure design settings](#)

[Shape report data](#)

[Filter data](#)

[Group data](#)

[Sort data](#)

[Format data](#)

Interactive features

[Hyperlink](#)

[Drilldown](#)

[Interactive sorting](#)

Working with Expressions

Expressions are used to specify criteria for retrieving and formatting data, creating calculated fields and calculating summaries, conditionally shaping data and changing a report control's appearance.

[Expressions builder](#)

[Open report](#)

[Open report](#)

[Save report](#)

[Save report](#)

[Preview report](#)

[Preview report](#)

Manage Schedule

This section explains on how to add, edit, delete schedules, manage report schedules and also on how to run the schedules on demand and enable or disable schedules in the Bold Reports Cloud.

[Manage Report Schedule](#)

[Schedule Report Settings](#)

Synchronization Schedule

Scheduling the synchronization of users and groups from Azure Active Directory, Database with the users and groups in the Bold Reports Cloud.

[Synchronization Schedule](#)

Notifications

You can configure the notifications settings to notify the users when any comment is added to the reports in the Bold Reports Cloud.

Notifications can be configured by both the System Administrator and user.

[Notification Settings](#)

Cloud Settings

The Bold Report Cloud Settings is used to configure the Azure Active Directory, Database Settings to import users and groups and synchronize their details after importing into the Bold Reports Cloud.

[Azure Active Directory](#)

[Database Settings](#)

Report Settings

User can able to make the report as public only if the system Administrator enable the **Mark report as public** option

[Report Settings](#)

Subscription

User can able to manage the subscription plan and also view the Plane Details and Subscription Information.

User can also cancel the subscription and activate new plane based on the requirement.

![Subscription](/static/assets/cloud/images/settings/subscription.png)

REST API

Using the Bold Reports Cloud REST API, you can manage and change Bold Reports resources programmatically via HTTP. The API gives you simple access to the functionality behind the resources on a Bold Reports. You can use this access to create your own custom applications or to script interactions with Bold Reports resources.

Frequently asked questions

The following topics which are discussed in this section will help you to get the answer for the frequently asked questions in **Bold Reports Cloud**.

- [List of IP addresses that need to be white listed for Bold Reports Cloud remote database access](#)
- [Does Report Server store reports execution data to the server database](#)

List of IP addresses that need to be white listed for Bold Reports Cloud remote database access

You should White list the following IP addresses on database servers to have a database access from Bold Reports Cloud.

- 52.170.46.128
- 52.170.43.17
- 52.170.46.174
- 52.170.41.18
- 13.72.72.252
- 52.170.41.233
- 52.186.122.173

Does Report Server store reports execution data to the server database

Bold Reports Cloud stores the following information only to the server. It does not store the reports execution data to the server database.

- Reports
- Shared data sources
- Shared data sets
- Categories
- Schedules definitions
- Permissions settings associated for report, data source, data set, categories, and schedules
- Report Server configurations

Overview

The Bold Reports is an online solution for analyzing data in web using the interactive reports with key metrics and tracking your business easily.

It is an end-to-end solution for creating, managing, and sharing interactive business report that includes a powerful report designer for composing easily.

Key features

Report management --- Reports can be efficiently organized under the categories. Permission to view the reports can be given to specific users or groups.

Business user friendly: The drag-and-drop friendly report designer allows business users to compose and publish reports without any help from IT.

Cost effective licensing: The licensing cost is effective for both small and large teams. Check the [pricing plans](#)

Multi-tenant: Multi-tenant mode provides a convenient, cost-effective method for deploying our report solution to multiple tenants.

Versioning --- All items stored in the Report Server are versioned, so it is possible to revert to an older version.

User management --- Users can be easily organized into groups to accurately map the structure of the small and large organizations.

Scheduling --- Reports can be generated and emailed according to a schedule. The scheduling functionality is very flexible.

Flexible permissions --- A flexible permission scheme controls the access to read, write, and delete reports.

View reports --- The built-in HTML 5 RDL Report Viewer control allows the user to view reports through the browser.

Export --- The RDL reports can be exported to Excel, Word, PDF, and HTML file formats.

Custom branding --- The Report Server has built-in customization capabilities that allows you to add your organization's name, logo, welcome note, etc.

Create a support incident

Still, if you are not able to find the information that you are looking for Report Server in the self-help resources mentioned above, please [contact us](#) by creating a support ticket.

Overview

The Bold Reports is an online solution for analyzing data in web using the interactive reports with key metrics and tracking your business easily.

It is an end-to-end solution for creating, managing, and sharing interactive business report that includes a powerful report designer for composing easily.

Key features

Report management --- Reports can be efficiently organized under the categories. Permission to view the reports can be given to specific users or groups.

Business user friendly: The drag-and-drop friendly report designer allows business users to compose and publish reports without any help from IT.

Cost effective licensing: The licensing cost is effective for both small and large teams. Check the [pricing plans](#)

Multi-tenant: Multi-tenant mode provides a convenient, cost-effective method for deploying our report solution to multiple tenants.

Versioning --- All items stored in the Report Server are versioned, so it is possible to revert to an older version.

User management --- Users can be easily organized into groups to accurately map the structure of the small and large organizations.

Scheduling --- Reports can be generated and emailed according to a schedule. The scheduling functionality is very flexible.

Flexible permissions --- A flexible permission scheme controls the access to read, write, and delete reports.

View reports --- The built-in HTML 5 RDL Report Viewer control allows the user to view reports through the browser.

Export --- The RDL reports can be exported to Excel, Word, PDF, and HTML file formats.

Custom branding --- The Report Server has built-in customization capabilities that allows you to add your organization's name, logo, welcome note, etc.

Create a support incident

Still, if you are not able to find the information that you are looking for Report Server in the self-help resources mentioned above, please [contact us](#) by creating a support ticket.

Overview

The Bold Reports On-Premise Edition is an end-to-end solution for creating, managing and sharing interactive business reports. It includes a powerful report server application for easily composing, managing and sharing reports.

Prerequisites

This section explains the system requirements to run Bold Reports On-Premise Edition.

Hardware Requirements

The following hardware requirements are necessary to run the Bold Reports On-Premise Edition:

- **Operating System :** Windows Client OS 8+ , Windows Server OS 2012 R2+
- **CPU :** 2-core, 2.4 GHz or faster, 32 bit or 64 bit processor.
- **Memory :** 8 GB RAM for 32 bit or 64 bit.
- **Hard drive :** 1.2 GB of free space (only installation files)

Software Requirements

The following software requirements are necessary to run the Bold Reports On-Premise Edition:

- **Framework :** [Microsoft .NET Framework 4.5](#)
- **Database :** Microsoft SQL Server 2008+, PostgreSQL server 9.0+
- **Web Server :** [Internet Information Services \(IIS\) 7.0+](#)
- **Web Browser :** Microsoft Edge , Mozilla Firefox , Chrome

Registration & Download

Please follow the below steps to install the Bold Reports On-Premise build.

1. Go to BoldReports [link](#) to register and download Bold Reports On-premise build.
2. Select On-Premise from Product Category.

![Bold Reports On-Premise Plans](/static/assets/on-premise/images/getting-started/trail-page.png)

3. Choose your plan based on your requirements.

![Bold Reports Trail Plans Page](/static/assets/on-premise/images/getting-started/trail-plans.png)

4. Create your Bold Reports On-premise portal by creating new account or sign in with existing account.

![Bold Reports Portal Registration](/static/assets/on-premise/images/getting-started/portal-registration.png)

5. Once portal have been created, you will be redirected to your trail downloads page.

![Bold Reports download option](/static/assets/on-premise/images/getting-started/download-option.png)

6. Click on download button and then select the file type to download.

![Bold Reports download setup](/static/assets/on-premise/images/getting-started/download-setup.png)

Installation and Deployment

This section explains on how to install and deploy the Bold Reports On-Premise Edition.

Installation

This topic details the steps required to install the Bold Reports On-Premise.

To learn about the system requirements needed to deploy the Bold Reports On-Premise in your business environment, see [System Requirements](#).

Run the Bold Reports On-Premise Installer and sign-in with your registered e-mail address to unlock the setup.

![installation with registered e-mail](/static/assets/on-premise/images/getting-started/installation-sign-in.png)

You can check the License Agreement of Bold Reports On-Premise Edition by clicking on the [License Terms and Conditions](#).

After you read the license agreement, click on Next to choose your portal license. This step is skipped if you have only one portal license

![portal selection](/static/assets/on-premise/images/getting-started/portal-plans-selection.png)

After you selected the portal license, click on Next to select the installation server type, location and the port number to host the Bold Reports On-Premise.

We have provided the Bold Reports On-Premise Edition to be hosted into the following two web server types

1. IIS Express
2. IIS

IIS Express

![Installation Location and IIS Express](/static/assets/on-premise/images/getting-started/installation-IISExpress.png)

IIS

Need to provide the Port number and Site Name to host the Bold Reports On-Premise into the IIS.

![Installation Location, IIS Port Changes and Site Name](/static/assets/on-premise/images/getting-started/installation-IIS.png)

After installation process is completed, you can launch the application by clicking on Launch Application.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/installation-launchapplication.png)

How to change the binding in the Bold Reports On-premise

Please follow below steps to change the application binding information

1. Add new binding to Bold Reports On-premise edition on IIS as shown in the image below.

Example: 192.168.1.3

![IIS Binding](/static/assets/on-premise/images/getting-started/add-binding.png)

Information: Don't remove existing bindings

2. Update the new binding values in below configuration files in deployed location.

Information: By default, Bold Reports will be deployed on C:\Bold Reports

- Update the **InternalAppReportUrl** value in config file in below location

{Deployed Location}\IDP\App_Data\Configuration\config.xml

![IDP Config File](/static/assets/on-premise/images/getting-started/idp-config.png)

- Update the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file in below location

{Deployed Location}\Report Server\App_Data\Configuration\Config.xml

![RS Config File](/static/assets/on-premise/images/getting-started/rs-config.png)

3. Restart the site in IIS and browse the site with old binding URL.
4. Now navigate to the site settings page of the UMS application using below old binding and update the new binding information as shown in image below.

http://localhost:{port-no}/ums/administration

![IDP Base URL](/static/assets/on-premise/images/getting-started/idp-base-url.png)

5. Now navigate to the site setting of your report server application using below old binding and update the new binding information as shown in image below.

http://localhost:{port-no}/reporting/en-us/site/site1/administration

![RS Base URL](/static/assets/on-premise/images/getting-started/rs-base-url.png)

6. Now Bold Reports site can be browsed using the new binding.

See Also

- [Application Startup](#)

Installation and deployment

This section explains how to install and deploy the Bold Reports On-Premise Edition.

Installation

This topic explains the steps required to install the Bold Reports On-Premise.

To learn about the system requirements needed to deploy the Bold Reports On-Premise in your business environment, see [System Requirements](#).

Run the Bold Reports On-Premise Installer and sign-in with your registered e-mail address to unlock the setup.

![installation with registered e-mail](/static/assets/on-premise/images/getting-started/installation-sign-in.png)

You can check the license agreement of Bold Reports On-Premise Edition by clicking the **License Terms and Conditions**.

After you read the license agreement, click Next to choose your portal license. This step is skipped, if you have only one portal license.

![portal selection](/static/assets/on-premise/images/getting-started/portal-plans-selection.png)

After selecting the portal license, click Next to select the installation server type, location, and the port number to host the Bold Reports On-Premise.

You have provided the Bold Reports On-Premise Edition to be hosted into the following two web server types:

1. IIS Express
2. IIS

IIS Express

![Installation Location and IIS Express](/static/assets/on-premise/images/getting-started/installation-IISExpress.png)

IIS

Need to provide the port number and site name to host the Bold Reports On-Premise into the IIS.

![Installation Location, IIS Port Changes and Site Name](/static/assets/on-premise/images/getting-started/installation-IIS.png)

After installation process is completed, you can launch the application by clicking the Launch Application.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/installation-launchapplication.png)

Bold BI with Bold Reports On-Premise Edition Installation

If Bold BI is already installed in your machine and installing Bold Reports On-Premise Edition, it will ask confirmation for Common Login.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/common-login.png)

Click **yes**, it will have common login for **BoldBI** and **BoldReportsOn-PremiseEdition** products. In IIS, it will deploy as single site.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/common-login-in-IIS.png)

Click **No**, it will not have common login for **BoldBI** and **BoldReportsOn-PremiseEdition** products. In IIS, it will deploy two sites **BoldBIOnPremiseEdition** and **BoldReportsOnPremiseEdition**.

[How to change the binding in the Bold Reports On-premise](#)

The Following steps to change the application binding information.

1. Add new binding to Bold Reports On-Premise Edition on IIS as shown in the following image.

Example: 192.168.1.3

![IIS Binding](/static/assets/on-premise/images/getting-started/add-binding.png)

Information: Don't remove existing bindings

2. Update the new binding values in following configuration files in deployed location.

Information: By default, Bold Reports will be deployed on C:\Bold Reports

- Update the **InternalAppReportUrl** value in config file in below location

{Deployed Location}\IDP\App_Data\Configuration\config.xml

![IDP Config File](/static/assets/on-premise/images/getting-started/idp-config.png)

- Update the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file in below location

{Deployed Location}\Report Server\App_Data\Configuration\Config.xml

![RS Config File](/static/assets/on-premise/images/getting-started/rs-config.png)

3. Restart the site in IIS and browse the site with old binding URL.
4. Now navigate to the site settings page of the UMS application using the following old binding and update the new binding information as shown in the following image.

http://localhost:{port-no}/ums/administration

![IDP Base URL](/static/assets/on-premise/images/getting-started/idp-base-url.png)

5. Now navigate to the site setting of your report server application using below old binding and update the new binding information as shown in image below.

http://localhost:{port-no}/reporting/en-us/site/site1/administration

![RS Base URL](/static/assets/on-premise/images/getting-started/rs-base-url.png)

6. Now, Bold Reports site can be browsed using the new binding.

* To Configure the Bold Reports On-Premise Edition using newly added bindings in IIS, refer this [link](#)

* Bold Reports On-Premise Edition configuration completed using newly added bindings in IIS but unable to launch BoldReportsOn-Premise site, refer this [link](#) to solve the problem and configure BoldReports site.

See Also

- [Application Startup](#)

Installation and Deployment

This section explains on how to install and deploy the Bold Reports On-Premise Edition.

Installation

This topic details the steps required to install the Bold Reports On-Premise.

To learn about the system requirements needed to deploy the Bold Reports On-Premise in your business environment, see [System Requirements](#).

Run the Bold Reports On-Premise Installer and sign-in with your registered e-mail address to unlock the setup.

![installation with registered e-mail](/static/assets/on-premise/images/getting-started/installation-sign-in.png)

You can check the License Agreement of Bold Reports On-Premise Edition by clicking on the [License Terms and Conditions](#).

After you read the license agreement, click on Next to choose your portal license. This step is skipped if you have only one portal license

![portal selection](/static/assets/on-premise/images/getting-started/portal-plans-selection.png)

After you selected the portal license, click on Next to select the installation server type, location and the port number to host the Bold Reports On-Premise.

We have provided the Bold Reports On-Premise Edition to be hosted into the following two web server types

1. IIS Express
2. IIS

IIS Express

![Installation Location and IIS Express](/static/assets/on-premise/images/getting-started/installation-IISExpress.png)

IIS

Need to provide the Port number and Site Name to host the Bold Reports On-Premise into the IIS.

![Installation Location, IIS Port Changes and Site Name](/static/assets/on-premise/images/getting-started/installation-IIS.png)

After installation process is completed, you can launch the application by clicking on Launch Application.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/installation-launchapplication.png)

[Bold BI with Bold Reports On-Premise Edition Installation](#)

If Bold BI already installed in your machine and your installing Bold Reports On-Premise Edition, it will ask confirmation for Common Login.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/common-login.png)

Click on yes, it will have common login for **BoldBI** and **BoldReportsOn-PremiseEdition** products. In IIS, it will deploy as Single site.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/common-login-in-IIS.png)

Click on **No**, it will not have common login for **BoldBI** and **BoldReportsOn-PremiseEdition** products. In IIS, it will deploy two sites **BoldBIOnPremiseEdition** and **BoldReportsOnPremiseEdition**.

Bold Reports Azure App Service can be deployed in Azure by following the [link](#).

How to change the binding in the Bold Reports On-premise

Please follow below steps to change the application binding information

1. Add new binding to Bold Reports On-premise edition on IIS as shown in the image below.

Example: 192.168.1.3

![IIS Binding](/static/assets/on-premise/images/getting-started/add-binding.png)

Don't remove existing bindings.

2. Update the new binding values in below configuration files in deployed location.

Information: By default, Bold Reports will be deployed on **C:\Bold Reports**

- Update the **InternalAppReportUrl** value in config file in below location

{Deployed Location}\IDP\App_Data\Configuration\config.xml

![IDP Config File](/static/assets/on-premise/images/getting-started/idp-config.png)

- Update the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file in below location

{Deployed Location}\Report Server\App_Data\Configuration\Config.xml

![RS Config File](/static/assets/on-premise/images/getting-started/rs-config.png)

3. Restart the site in IIS and browse the site with old binding URL.
4. Now navigate to the site settings page of the UMS application using below old binding and update the new binding information as shown in image below.

http://localhost:{port-no}/ums/administration

![IDP Base URL](/static/assets/on-premise/images/getting-started/idp-base-url.png)

5. Now navigate to the site setting of your report server application using below old binding and update the new binding information as shown in image below.

http://localhost:{port-no}/reporting/en-us/site/site1/administration

![RS Base URL](/static/assets/on-premise/images/getting-started/rs-base-url.png)

6. Now Bold Reports site can be browsed using the new binding.

* To Configure the Bold Reports On-Premise Edition by using newly added bindings in IIS, refer this [link](#)

* Bold Reports On-Premise Edition configuration completed by using newly added bindings in IIS but unable to launch **BoldReportsOn-Premise** site, refer this [link](#) to solve the problem and configure BoldReports site.

See Also

- [Application Startup](#)

Upgrade Bold Reports On-Premise version from 1.x to latest

This section explains how to upgrade Bold Reports Server from any version to 2.x version.

Bold Reports releases several major versions in a year. Each version includes new features, bug fixes and other improvements.

Bold Report Server can be upgraded to version 2.x at any time manually, and there are no automatic updates for Report Server. Before upgrading, you can refer the features and enhancements from the [Release Notes](#).

Steps to upgrade Bold Reports On-Premise

- Before installing the Bold Reports version 2.x or higher, make sure to take Backup of the database files from the below location.

C:\Bold Reports\Report Server\ReportServer.Web\App_Data

C:\Bold Reports\UMS\UMS.Web\App_Data

- Once done, uninstall the current Bold Report Server.
- Install the latest Bold Report Server by following this [link](#) and configure the application using SQL Database by following this [link](#).
- Once configuration is done, you can migrate your old data to this application by following the [document](#).

Upgrade Bold Reports On-Premise version from 1.x to latest

This section explains how to upgrade Bold Reports Server from any version to 2.x version.

Bold Reports releases several major versions in a year. Each version includes new features, bug fixes and other improvements.

Bold Report Server can be upgraded to version 2.x at any time manually, and there are no automatic updates for Report Server. Before upgrading, you can refer the features and enhancements from the [Release Notes](#).

Steps to upgrade Bold Reports On-Premise

- Before installing the Bold Reports version 2.x or higher, make sure to take Backup of the database files from the below location.

C:\Bold Reports\Report Server\ReportServer.Web\App_Data

C:\Bold Reports\UMS\UMS.Web\App_Data

- Once done, uninstall the current Bold Report Server.
- Install the latest Bold Report Server by following this [link](#) and configure the application using SQL Database by following this [link](#).
- Once configuration is done, you can migrate your old data to this application by following the [document](#).

Migrate Bold Reports On-Premise

This section explains how to migrate your resources from v4.1 (Syncfusion Reports Server) or v1.2 (Bold Reports Single Tenant) to v2.1 (Bold Reports Multi Tenant).

Prerequisites

- Take backup of the database and resources files from the below folder location.

| Product | Location | Back Up Example |
|------------------------------------|-----------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| Syncfusion Report Server | C:\Syncfusion\ReportServer\ReportServer.Web\AppDataC:\Syncfusion\UMS\UMS.Web\AppData | C:\backupsyncfusion\ReportServer\AppDataC:\backupsyncfusion\UMS\AppData |
| Bold Report Server (Single Tenant) | C:\Bold Reports\ReportServer\ReportServer.Web\AppDataC:\BoldReports\UMS\UMS.Web\AppData | C:\backupboldreports\ReportServer\AppDataC:\backupboldreports\UMS\AppData |

For an Azure App Service, you have to take a backup of App_Data folder by using the FTP application and place the backup folder in the machine which you installed the bold reports v2.1.

- After taken the backup, uninstall the current Syncfusion / Bold Report Server.
- Install the latest Bold Reports Server as fresh installation by following this [link](#)
- Configure the application using SQL Database by following this [link](#) and while configuring the admin details you must use the admin email-id of Syncfusion Reports Server / Bold Reports Server (Single Tenant).
- Once configuration is done, you can migrate your old data to this application.

Steps to run the data migration utility

- Download the migration tool from this [link](#).
- Extract the file and edit the Syncfusion.Server.DataMigration.exe.config file.
- You need to update the values of below listed keys in the config file.

-
-
-

![data-migration-config](/static/assets/on-premise/images/data-migration/migration-config.png)

4. Copy the Configuration path in your Report Server back up folder and paste it to the value of ReportsBackUpConfigPath key. For example, .
5. Go to the installation location C:\Bold Reports\IDP\ and copy the decryptionkey value from web.config file. Then paste it to the value of IDPDecryptionKey key in Syncfusion.Server.DataMigration.exe.config. For example,

![idp-decryption-key](/static/assets/on-premise/images/data-migration/idp-decryption-key.png)

6. By default the Bold Reports installation will be deployed under C:\Bold Reports\IDP and the same will be updated in Syncfusion.Server.DataMigration.exe.config file. If the Bold Reports deployed in some other location means then you must update the IDP path location in the value of BoldReportsIDPPPath key. For example,

![idp-location](/static/assets/on-premise/images/data-migration/idp-location.png)

7. Run the tool Syncfusion.Server.DataMigration.exe from the extracted location.

![data-migration-tool](/static/assets/on-premise/images/data-migration/migration-exe.png)

* The migration tool can migrate your resources like Reports, Data Sets, Data Sources, Schedules, Users, Groups and Permissions.

* Also, the utility will not migrate the following items like User's without email address, Admin Settings and Profile Pictures.

* If you need any further assistance please contact Syncfusion Support.

Upgrade Bold Reports On-Premise version from 2.x to latest

This section explains how to upgrade Bold Reports Server from 2.x to latest.

Bold Reports releases several major versions in a year. Each version includes new features, bug fixes and other improvements.

Bold Report Server can be upgraded to latest version at any time manually, and there are automatic updates for Report Server. Before upgrading, you can refer the features and enhancements from the [Release Notes](#).

Upgrading guidelines

Bold Reports recommends you to follow below guidelines while upgrading the Bold Reports Server from an older version to latest version.

- Save all the open settings and the unsaved items.
- Ensure no one is currently working with reports.

- Inform about the maintenance time to the users.
- For databases, make sure you have a valid network connection to the database while upgrading to the new version.

Backup the existing data

Before upgrading the Bold Reports to latest version, make sure to take the backup of all the files and folders from below location.

| | |
|--------------|--------------------------------------------|
| Bold IDP | {Deployed Location}\IDP\App_Data |
| Bold Reports | {Deployed Location}\Report Server\App_Data |

By default, Bold Reports will be deployed in "C:\Bold Reports\"

Proceeding with upgrade

Bold Reports updates the database schema of your current version to the latest version. The upgrade process will retain all the resources and settings from the previous deployment.

You can download the latest version of the Bold Reports On-Premise Edition under your account [here](#).

Steps to upgrade Bold Reports On-Premise

Run the Bold Reports On-Premise Installer and sign-in with your registered e-mail address to unlock the setup.

![installation with registered e-mail](/static/assets/on-premise/images/getting-started/upgrade-install-setup.png)

You can check the License Agreement of Bold Reports On-Premise Edition by clicking the **License Terms and Conditions** and click on **Next** button.

Then, upgrading prompt will be shown, click **Yes** button to proceed the installation.

![Upgrade Installation Prompt](/static/assets/on-premise/images/getting-started/upgrade-prompt.png)

Click **Install** to continue upgrading the Bold Reports On-Premise using the existing server hosting type, location, and the port number.

The installer will automatically detect the existing hosting details and show in any one of the following web hosting types:

1. IIS Express
2. IIS

![Installation Location, IIS Port Changes and Site Name](/static/assets/on-premise/images/getting-started/upgrade-web-server-type.png)

After installation process is completed, you can launch the application by clicking **Launch Application**.

![Installation LaunchApplication](/static/assets/on-premise/images/getting-started/upgrade-launch-app.png)

Azure

Bold Reports Enterprise Edition offers to create Report Server with Azure App Service using ARM template. This will helpful to Easily scale up your Report Server application on demand by deploying it as an Azure App Service.

You can refer the below documentation for creating and upgrading your Azure App Service.

[Create Bold Reports Azure App Service using ARM template](#)

[Upgrade Bold Reports Azure App Service from v1.x to latest](#)

[Upgrade Syncfusion Report Server Azure App Service from v4.x to latest](#)

Create a Bold Reports Azure App Service using the ARM template

If you already having Report Server Azure App Service with Syncfusion Report Platform Report Server or Bold Reports v1.x Report Server, then refer to the Syncfusion Report Server migration or upgrade v1.x for your startup.

This section explains how to create the Bold Reports Report Server into Azure cloud as App Service using the ARM template by following these steps:

1. Log in to [Azure portal](#).
2. Click **Create resource** from the top menu.

![New in Azure portal](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/new-template.png)

3. Search **Template deployment** in the marketplace and select **Template deployment**.

![Search Template deployment option](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/search-template.png)

4. Click **Create**.

![Create Template deployment](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/create-template.png)

5. Now, select **Build your own template in the editor** in the Custom Deployment window.

![Build your own template in the editor option](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/build-your-own-template.png)

6. [Click here](#) to download the ARM template file.
7. Copy all the contents in the template file and replace them in the Edit template window, and then click **Save**.

![Edit the template and replace the template](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/edit-template.png)

8. Fill the details as instructed in the following to complete the deployment steps.
 - **Subscription:** Choose the subscription that you have with Azure. Learn more about subscriptions from [here](#).
 - **Resource group:** This is a logical group in Azure to group your resources like web app, storage account, network etc. Learn more about resource groups from [here](#).
 - **Location:** Choose the location to deploy the app. **East US** is our recommended location.
 - **Web App name:** This is the name of the Report Server that you want to have in the URL. As this is going to be the URL, it should be unique globally. Deployment process will be failed, if this is already present and you should start once again with another name.
 - **Storage account name:** This is mandatory for Blob storage and also must be unique as that of the Web App name. Learn more about storage accounts from [here](#).
 - **Storage account type:** This is optional, if you have chosen file storage in storage type and mandatory for Blob storage. Learn more about storage account types from [here](#).
9. Select the agreement check box and click **Purchase** to deploy the Bold Reports On-Premise Web App.

![Fill app service details](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/custom-template-details-form.png)

Now, Bold Reports Multi-tenant App Service(Web App) deployment get started.

The App Service plan for the Web App is created, which will be in **Basic – B1** by default. Learn more about App Service plans from [here](#). Bold Reports On-Premise Web App does not support free or shared App Service plans.

Bold Reports On-Premise supports basic, standard, and premium App Service plans in the Azure. The minimum recommended App Service plan to run the application is the basic plan.

To get better performance, scale up the App Service plan from basic to standard or premium plans. Refer to the following documentation links to learn how to scale up and scale out the App Service plan.

Scale up: <https://docs.microsoft.com/en-us/azure/app-service-web/web-sites-scale>

Scale out: <https://docs.microsoft.com/en-us/azure/monitoring-and-diagnostics/insights-how-to-scale>

10. After the successful deployment go to your Web App and search **App Service Editor**, then click **GO** as shown in the following screenshot.

![App Service Editor](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/app-service-editor.png)

11. Copy your Web App URL, and then update the following xml node **InternalAppReportUrl** in the config.xml file under **wwwroot/IDP/UMS/Configuration/boldreports/config.xml**.

The URL must be, **InternalAppReportUrl- {web App URL}/reporting**

![IDP Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/idp-config.png)

12. Copy your Web App URL, and then update the following xml node `InternalAppDataServiceUrl` and `InternalAppIdpUrl` in the config.xml file under `wwwroot/ReportServer/Configuration/config.xml`.

The URL must be, `InternalAppDataServiceUrl – {web App URL}/reporting/reportservice`

`InternalAppIdpUrl – {web App URL}`

![Report Server Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/report-server-config.png)

Now, stop and start your Web App, and then browse it.

Configure a new Bold Reports

Configure the Bold Reports Multi-tenant Azure App Service by following these steps:

1. Click the `Proceed to Setup`.

![Proceed to Setup](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/proceed-to-setup-button.png)

2. Provide your [Database details](#) to maintain your user identities and Bold Reports tenants and click `Next`.

![IDP Database Configuration](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/idp-database-configuration.png)

3. Provide the [user details](#) and this user will be an admin for Bold Reports, and then click `Next`.

![Admin Details](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/admin-details.png)

If you wish to migrate your Syncfusion Report Server Azure app service(v4.x) data or Bold Reports Azure app service version(v1.x) data to Bold Reports Multi-tenant Azure App Service, while configuring the admin details you must use the admin email id of Syncfusion Reports Server or Bold Reports Server (Single Tenant).

4. Provide your [Database details](#) to maintain your reports, users, and their access permissions, and then click `Next`.

![Report Server Database Configuration](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/report-server-database-configuration.png)

5. Click the check box to fetch the pre-configured blob details at the deployment time to maintain your physical files of reports, datasets, data sources etc, and then click `Next`.

![Azure Blob Configuration](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/azure-blob-configuration.png)

- Once the configuration is done, then you can able to navigate sites or reports page by success login.

Upgrade Bold Reports Azure App Service version from v1.x to latest

This section explains how to upgrade Bold Reports Azure app Service from 1.x to latest.

Create a Bold Reports Azure App Service with latest version

- Create the Bold Reports Report Server Azure App Service by following this [link](#) and also configure the application using the database by following this [link](#).

File storage resources for migration

- Create the folder **IDP** and **Report Server** in your local machine as shown in the following screenshot.

![Report Server Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/folder-in-local-machine.png)

- Connect your Bold Report Server (single-Tenant) Web App by FileZilla, and then by dragging you can move the **App_Data** folder to **Report Server** folder, which you have created.

![Report Server App_Data](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/report-server-app-data.png)

- Create the following sub-folders structure like **/App_Data/Configuration/** under **IDP** folder.
- Go to your storage account, which is created at the deployment time of Bold Reports Multi-tenant and in container section navigate to the following location.

boldreportscontainer/tenantmanagement/tenant management server/configuration

- Download the config.xml file to the folder **IDP/App_Data/Configuration**.

![Download Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/download-config.png)

- Connect your Bold Reports Multi-tenant Azure app by FileZilla, and then drag the **Privatekeys.dat** file to the local machine under the folder **IDP/App_Data/Configuration/**.

![Key file](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/key-file.png)

- Run the [data migration utility](#).

Upgrade Bold Reports Azure App Service version from v1.x to latest Azure Blob Storage resources for migration

Azure Blob Storage resources for migration

1. Create the folder **IDP** and **Report Server** in your local machine as shown in the following screenshot.

![Report Server Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/folder-in-local-machine.png)

2. Create the following sub-folders structure like **/App_Data/Configuration/** under **IDP** folder.
3. Go to your storage account, which is created at the deployment time of Bold Reports Multi-tenant and in container section navigate to the following location.

boldreportscontainer/tenantmanagement/tenant management server/configuration

4. Download the config.xml file to the folder **IDP/App_Data/Configuration.**

![Download Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/download-config.png)

5. Connect your Bold Reports Multi-tenant Azure app by FileZilla, and then drag the **Privatekeys.dat** file to the local machine under folder **IDP/App_Data/Configuration/**.

![Key file](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/key-file.png)

6. Create the following sub-folders structure like **/App_Data/Configuration/** under **Report Server** folder in your local machine.
7. Go to the storage account of previous Report Server (Single-Tenant) and in container section navigate to the following location.

boldreportsrscontainer/Bold Reports/Report Server/Configuration/

8. Download the config.xml file to the folder **Report Server/App_Data/Configuration.**

![Download Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/download-config-file.png)

9. Run the [data migration utility](#).

Data migration with migration utility

1. Download the migration tool from this [link](#).
2. Extract the file and open the **Syncfusion.Server.DataMigration.exe.config** file. You need to update the values to the following listed keys in the config file:

```
<add key="ReportsBackUpConfigPath" value="" />  
<add key="BoldReportsIDPPPath" value="" />
```

3. Copy the **IDP** path and paste it to the value of **BoldReportsIDPPPath** key.

For example, `<add key="BoldReportsIDPPPath" value="D:\migration\IDP"/>`

4. Copy the **Configuration** path of the Report Server and paste it to the value of **ReportsBackUpConfigPath** key.

For example, `<add key="ReportsBackUpConfigPath" value="D:\migration\ReportServer\App_Data\Configuration"/>`

![Key file](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/app-settings-key.png)

5. Run the tool `Syncfusion.Server.DataMigration.exe` from the extracted location.

![DataMigration Exe](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/data-migration-exe.png)

The migration tool will migrate the users, groups, permissions, reports, data sources, datasets, and schedules. The user without email address is skipped, because for the Bold Reports Report Server email is mandatory.

Migrate Syncfusion Report Platform Report Server Azure App Service

This section explains how to upgrade the Syncfusion Report Server Azure App Service to Bold Reports.

You should have Syncfusion Report Server Azure App Service in version 4.x

Create a Bold Reports Azure App Service

- Create the Bold Reports Report Server Azure App Service by following this [link](#) and also configure the application using the database by following this [link](#).

File storage resources for migration

1. Create the folder **IDP** and **Report Server** in your local machine as shown in the following screenshot.

![Report Server Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/folder-in-local-machine.png)

2. Connect your Syncfusion Report Server Web App by FileZilla, and then by dragging you can move the **App_Data** folder to **Report Server** folder, which you have created.

Migrate Syncfusion Report Platform Report Server Azure App Service Azure Blob Storage resources for migration

![Report Server App_Data](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/report-server-app-data.png)

3. Create the following sub-folders structure like **/App_Data/Configuration/** under **IDP** folder.
4. Go to your storage account, which is created at the deployment time of Bold Reports and in container section navigate to the following location.

boldreportscontainer/tenantmanagement/tenant management server/configuration

5. Download the config.xml file to the folder **IDP/App_Data/Configuration.**

![Download Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/download-config.png)

6. Connect your **Bold Reports (Multi-tenant)** Azure app by FileZilla, and then drag the **Privatekeys.dat** file to the local machine under the folder **IDP/App_Data/Configuration/.**

![Key file](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/key-file.png)

7. Run the [data migration utility](#).

Azure Blob Storage resources for migration

1. Create the folder **IDP** and **Report Server** in your local machine as shown in the following screenshot.

![Report Server Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/folder-in-local-machine.png)

2. Create the following sub-folders structure like **/App_Data/Configuration/** under **IDP** folder.
3. Go to your storage account, which is created at the deployment time of Bold Reports and in container section navigate to following location.

boldreportscontainer/tenantmanagement/tenant management server/configuration

4. Download the config.xml file to the folder **IDP/App_Data/Configuration.**

![Download Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/download-config.png)

5. Connect your **Bold Reports (Multi-tenant)** Azure app by FileZilla, and then drag the **Privatekeys.dat** file to the local machine under folder **IDP/App_Data/Configuration/.**

![Key file](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/key-file.png)

6. Create the following sub-folders structure like `/App_Data/Configuration/` under `Report Server` folder in your local machine.
7. Go to the storage account of previous Report Server (Single-Tenant) and in container section navigate to the following location.

`boldreportsrscontainer/Bold Reports/Report Server/Configuration/`

8. Download the config.xml file to the folder `Report Server/App_Data/Configuration.`

![Download Config](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/download-config-file.png)

9. Run the [data migration utility](#).

Data migration with migration utility

1. Download the migration tool from this [link](#).
2. Extract the file and open the `Syncfusion.Server.DataMigration.exe.config` file. You need to update the values to the following listed keys in the config file:

```
<add key="ReportsBackUpConfigPath" value="" />
<add key="BoldReportsIDPPath" value="" />
```

3. Copy the `IDP` path and paste it to the value of `BoldReportsIDPPath` key.

For example, `<add key="BoldReportsIDPPath" value="D:\migration\IDP"/>`

4. Copy the `Configuration` path of the Report Server and paste it to the value of `ReportsBackUpConfigPath` key.

For example, `<add key="ReportsBackUpConfigPath" value="D:\migration\Report Server\App_Data\Configuration"/>`

![Key file](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/app-settings-key.png)

5. Run the tool `Syncfusion.Server.DataMigration.exe` from the extracted location.

![DataMigration Exe](/static/assets/on-premise/images/installation-and-deployment/azure-deployment/app-service-using-arm-template/data-migration-exe.png)

The migration tool will migrate the users, groups, permissions, reports, data sources, datasets, and schedules. The user without email address is skipped, because for the Bold Reports Report Server email is mandatory.

Application Startup

This topic describes how to startup the Bold Reports On-Premise Application.

Application startup screen helps you to configure storage options and admin account setup.

Application startup holds the storage options configuration for below application,

1. Bold Reports Sites - Manages user identities and sites
2. Bold Reports Server- Manages reports, data sources, dataset, schedules and the user permissions.

Click **Proceed to Setup**.

![Proceed to Setup](/static/assets/on-premise/images/getting-started/proceed-to-setup.png)

Database Configuration for Bold Reports Sites

This configuration stores the user identities and sites details in SQL Server Database or PostgreSQL.

You can connect to the existing SQL Server instance with the below options.

- Create new database.
- Use an existing database.

![SQL Server](/static/assets/on-premise/images/getting-started/application-startup-step1.png)

You can also connect to the existing PostgreSQL Server instance with the below options.

- Create new database.
- Use an existing database.

![PostgreSQL Server](/static/assets/on-premise/images/getting-started/application-startup-postgreSQL-server.png)

The credentials that is given to connect to the SQL Server or PostgreSQL Server instance must have permissions to

- * Create Database
- * Create Table
- * Insert
- * Update Table
- * Alter Table
- * Select
- * Drop Table
- * Drop Database

Storage Type

User can select the preferred storage type File Storage or Blob Storage to store the resource in BoldReports Sites

![Storage Type](/static/assets/on-premise/images/getting-started/storage-type.png)

Blob Storage

User can select the Blob Storage by giving Azure Blob Credential in the required field

![Blob Storage](/static/assets/on-premise/images/getting-started/select-storage-type-blobstorage.png)

New User - System Administrator

New user should be created to access the Bold Reports Sites and Reports with the details mentioned in the below image.

![Admin User Registration](/static/assets/on-premise/images/getting-started/application-startup-admin.png)

While creating this new user account, a new group **System Administrator** is also created.

By default, **System Administrator** group have permission to do the below

- Create Reports
- Create Data Sources
- Create Dataset
- Create Schedules
- Create Users
- Create Groups
- Manage Permissions for users and groups

The new user account created is assigned to this group by default.

![Bold Reports Server](/static/assets/on-premise/images/getting-started/starting-bold-reports-server.png)

Database Configuration for Bold Reports Server

This configuration stores the reports, users and their access permissions in SQL Server Database or PostgreSQL Server Database.

You can connect to the existing SQL Server instance with the below options.

- Create new database.
- Use an existing database.

![SQL Server Site](/static/assets/on-premise/images/getting-started/application-startup-site-registration.png)

You can connect to the existing PostgreSQL Server instance with the below options.

- Create new database.
- Use an existing database.

![PostgreSQL Server Site](/static/assets/on-premise/images/getting-started/application-startup-site-registration-postgreSQL-server.png)

The credentials that is given to connect to the SQL Server instance must have permissions to

| Azure Active Directory Settings | Storage Type |
|---------------------------------|--------------|
|---------------------------------|--------------|

- * Create Database
- * Create Table
- * Insert
- * Update Table
- * Alter Table
- * Select
- * Drop Table
- * Drop Database

Storage Type

User can select the preferred storage type File Storage or Blob Storage to store the report,datasource and dataset in BoldReports Server.

![Storage Type](/static/assets/on-premise/images/getting-started/storage-type.png)

Once the process completed, you can able to

- Manage sites – where you can manage permissions for the sites you have created.
- Go to Reports - where you can create, edit, update the reports and so on.

![Launch Bold Reports Reports](/static/assets/on-premise/images/getting-started/launch-application.png)

Azure Active Directory Settings

You should configure the Azure Active Directory settings to import users and groups from the Azure Active Directory and synchronize their details after importing into the Bold Reports On-Premise. To configure the [Azure Active Directory](#) with the Bold Reports On-Premise, follow these steps:

Click the settings option in the Report Server left-side panel and navigate to USER DIRECTORY tab, and then select **Azure Active Directory** as shown in the following image.

![Azure Active Directory Settings](/static/assets/on-premise/images/settings/azure-active-directory-settings.png)

The tenant name, client ID, and client secret code are required to configure the Bold Reports On-Premise with Azure Active Directory.

Refer to [how to setup Azure Active Directory](#) section to get the following details from your Azure account:

- Tenant name: Default domain name of your Active Directory.
- Client ID: Client ID of the Report Server application created in your Azure Active Directory.
- Client secret code: Secure key of the Report Server application, which is created in your Azure Active Directory.

| | |
|-------------------------------------|--------------|
| Configure Active Directory settings | Storage Type |
|-------------------------------------|--------------|

Configure Active Directory settings

You should configure the Active Directory settings to import users and groups from the Windows Active Directory and synchronize their details after importing into the Bold Reports On-Premise. To configure [Active Directory](#) with the Bold Reports On-Premise, follow these steps:

1. Click the setting option in the Report Server left-side panel and navigate to the USER DIRECTORY tab, and then select **Windows Active Directory** as shown in the following image.

![Active Directory Settings](/static/assets/on-premise/images/settings/windows-active-directory-settings.png)

2. Set the following Active Directory details of your Windows Active Directory.
 - **Username and Password:** Username and password of the user present in the Active Directory domain.
 - **LDAP URL:** IP or DNS name of the Windows Server, where you have Active Domain Services enabled. For example, `LDAP://192.168.1.1`.
 - **Distinguished Name:** FQDN of the Active Directory. Follow the given procedure to find out the distinguished name of your Active Directory:
3. Go to Run(Win + R) and type `sysdm.cpl` in a machine, which is connected to the Active Directory or in the Windows Server, where you have Active Directory Domain Services enabled.

![Execute sysdm.cpl command](/static/assets/on-premise/images/settings/run-sysdm-command.png)

2. The **System Properties** dialog will be opened, and you can find the distinguished name labeled as **Domain**.

![Distinguished Name - Domain](/static/assets/on-premise/images/settings/view-active-directory-user-domain.png)

For example, if your Domain is `www.example.com`, then you have to type as
`dc=www,dc=example,dc=com`.

- **Enable SSL:** Enable this check box, if your Windows Server needs SSL connection to connect to the Active Directory.
- **Port Number:** Default port for non SSL connections is 389 and for SSL connections is 636.
 3. Click **Test Connection** to validate the connection with given details and click **Save**.

Azure Active Directory Settings

You should configure the Azure Active Directory settings to import users and groups from the Azure Active Directory and synchronize their details after importing into the Bold Reports On-Premise. To configure the [Azure Active Directory](#) with the Bold Reports On-Premise, follow these steps:

Click the settings option in the Report Server left-side panel and navigate to USER DIRECTORY tab, and then select **Azure Active Directory** as shown in the following image.

![Azure Active Directory Settings](/static/assets/on-premise/images/settings/azure-active-directory-settings.png)

Email Settings

Configure database connection details

The tenant name, client ID, and client secret code are required to configure the Bold Reports On-Premise with Azure Active Directory.

- Tenant name: Default domain name of your Active Directory.
- Client ID: Client ID of the Report Server application created in your Azure Active Directory.
- Client secret code: Secure key of the Report Server application, which is created in your Azure Active Directory.

Email Settings

The Report Server includes an email delivery settings to perform the following operations:

- Account activation: Sends user account activation email.
- Forgot password: Sends request links to reset the password when users forget the password.
- Reset password: Sends links to reset the password.
- Schedule and deliver reports: Sends the exported report to the scheduled recipients.

The email delivery works with the [SMTP](#) mail server technology.

The Report Server email delivery extension is not configured by default.

To configure the email settings, click the setting option in the Report Server left-side panel and navigate to the Email tab as shown in the following image.

![Email settings page](/static/assets/on-premise/images/settings/email-settings.png)

The following SMTP details are required to send email from the Report Server:

- SMTP server: Specifies the remote SMTP server or forwarder.
- SMTP port: Configures for port 25.
- Sender name: Sets the value that appears in the From: line of an email message.
- Sender email: Sets the value of the sender email.
- Authentication type: Specifies the connection of the Report Server connects with the remote SMTP Server.
- Password: Specifies the password to connect to the SMTP Server.
- SSL: Specifies whether the `SmtpClient` uses Secure Sockets Layer (SSL) to encrypt the connection.

Database settings

You should configure the database settings in the Report Server to import users and groups from an existing database.

Configure database connection details

1. To configure the database connection details, click the settings option in the Report Server left-side panel.
2. Navigate to the User Directory tab and select **Database** as shown in the following image.

![Database connection configuration page](/static/assets/on-premise/images/settings/database-settings-page.png)

3. You can import users from the following types of databases:
 - o SQL Server
 - o MySQL
 - o Oracle
 - o PostgreSQL
4. Click **database type** drop-down to change the database type.

![Select database Type](/static/assets/on-premise/images/settings/select-database-type.png)

SQL Server database

1. Fill the SQL connection details such as server name, authentication type, username, password, and database name.
2. Click **Test Connection** to validate the connection details.

![Configuring SQL Server database connection](/static/assets/on-premise/images/settings/connect-to-sqlserver-database.png)

MySQL database

1. Fill the MySQL connection details such as DSN, username, password, and database name.
2. Click **Test Connection** to validate the connection details.

![Configuring MySQL database connection](/static/assets/on-premise/images/settings/connect-to-mysql-database.png)

Oracle database

1. Fill the Oracle connection details such as DSN, Admin username, Admin password, database name, and database password.
2. Click **Test Connection** to validate the connection details.

![Configuring Oracle database connection](/static/assets/on-premise/images/settings/connect-to-oracle-database.png)

PostgreSQL database

1. Fill the PostgreSQL connection details such as server name, port, username, password, and database name.
2. Click **Test Connection** to validate the connection details.

![Configuring PostgreSQL database connection](/static/assets/on-premise/images/settings/connect-to-postgresql-database.png)

Map database columns

1. After the connection details are filled, click **Next** to get schema of the provided database connection.

![Select Next option from database page](/static/assets/on-premise/images/settings/next-click.png)

2. The data page will be as follows, after the database has been connected.

![Database mapping column](/static/assets/on-premise/images/settings/select-columns-page.png)

3. Select the matched columns from the database schema for username, first name, email address, last name (optional), and **IsActive** (Optional) fields.

![Mapping database column with user fields](/static/assets/on-premise/images/settings/map-columns.png)

4. If the columns are selected from different tables, the following scenarios will happen:
 - o If the tables have single relationship - The join will be made with the particular relationship.
 - o If the tables have multiple relationships - The relationship can be selected from the list of relationships and then click **Apply**.

![Select table relationship](/static/assets/on-premise/images/settings/select-relation.png)

- If there is no relationship between tables - Form Join with the selected tables and then click **Apply**.

![Form table relationship](/static/assets/on-premise/images/settings/form-relation.png)

5. After selecting all the columns, click **Save** to save the database details. Based on this saved details, it will pull the users from the configured database.

![Saving database details](/static/assets/on-premise/images/settings/save-database-details.png)

6. After the changes are completed, the success message will be displayed as shown in the image.

![Database configuration success message](/static/assets/on-premise/images/settings/database-details-saved.png)

7. Use the Back button in the settings page to edit the database details before mapping and saving the database configuration.

Mark reports as public switch Allow/Restrict

This section explains how we can Allow/Restrict to mark the Reports as public from the Administration reports settings page.

we can able to control the user to make the Reports as public through the **Reports** settings tab in settings page. The user can able to mark the Reports as public, if the Administrator Allow the **Mark reports as public** option.

Take Control Over the Public Reports

If the Administrator has allowed, then the user can able to mark the Reports as public and other users can able to see the public Reports.

If the Administrator has restricted, the public Reports cannot be rendered by the user until they hold the permission for the Reports.

Mark reports as public switch Allow/Restrict Public Reports

After restricted the **Mark reports as public** option in reports settings page,

![Click on reportsettings icon](/static/assets/on-premise/images/settings/report-settings.png)

If click on **Make Public** option from context menu of the respective Reports, For administrator the following message will be shown

![Click on makepublic icon](/static/assets/on-premise/images/settings/makepublic-admin.png)

If click on **Make Public** option from context menu of the respective Reports, For user the following message will be shown

![Click on makepublic icon](/static/assets/on-premise/images/settings/makepublic-user.png)

After restricted **Mark reports as public** option, if click on **Public Reports** tab, for administrator the following message will be shown

![Click on publicreport icon](/static/assets/on-premise/images/settings/publicreport-admin.png)

After restricted the **Mark reports as public**, if click on **Public Reports** tab, for end user the following message will be shown

![Click on publicreport icon](/static/assets/on-premise/images/settings/publicreport-user.png)

Make Public

Follow the steps below to make the Reports accessible to anonymous users.

1. Allow the **Mark reports as public** option in reports settings page,

![Click on report-settings-on icon](/static/assets/on-premise/images/settings/report-settings-on.png)

2. Click on **Make Public** option from context menu of the respective Report,

![Click on makepublic icon](/static/assets/on-premise/images/settings/makepublic.png)

3. Click on **Make public** in the confirmation dialog box,then Reports will be marked as public.

Click [here](#) for more details about public reports.

OAuth 2.0 Support in Boldreports

OAuth 2.0 for Single sign-On (SSO) in BoldReports application, so that the user can login to BoldReports application after authenticating using the OAuth 2.0.

Prerequisites

1. An account with an OAuth 2.0 provider.
2. Register the BoldReports application in the OAuth 2.0 provider.

Steps to configure OAuth 2.0 in BoldReports

1. To configure the OAuth 2.0 connection details, click the settings option in the Report Server left-side panel.
2. Navigate to the Authentication tab and OAuth 2.0 as shown in the following image

![Authentication](/static/assets/on-premise/images/settings/authentication-settings.png)

3. Provide the following details in the OAuth 2.0 settings of BoldReports application.

| | |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authorization Endpoint | It is the endpoint in the provider to authorize the user. |
| Token Endpoint Method | It represents the request type to access the token endpoint. |
| Token Endpoint | It is the endpoint in the provider that generates the token. |
| User Information Endpoint Method | It is the endpoint in the provider used to get the user details. |
| User Information Endpoint | It represents the request type to access the user information endpoint. |
| Client ID | It is an unique identifier provided to each of the applications while registering in the providers. |
| Client Secret | It is a secret key that is used to authorize the applications. |
| Scopes | It is comma separated lists of identifiers that specifies the access privileges that are being requested from the provider. |
| Email | This must be the email of an admin account of the providers. |

![Active Directory Settings](/static/assets/on-premise/images/settings/oauth-authentication.png)

OpenID Connect support in BoldReports

OpenID Connect for Single sign-On (SSO) in BoldReports application, so that the user can login to BoldReports application after authenticating using the OpenID Connect

Prerequisites

1. An account with an OpenID Connect provider.
2. Register the BoldReports application in the OpenID Connect provider.

Steps to configure OpenID Connect in BoldReports

1. To configure the OpenID Connect connection details, click the settings option in the Report Server left-side panel.
2. Navigate to the Authentication tab and OpenID Connect as shown in the following image

![Active Directory Settings](/static/assets/on-premise/images/settings/authentication-settings.png)

3. Provide the following details in the OpenID Connect settings of BoldReports application.

| | |
|---------------|-----------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authority | It is the instance created for the user in the provider. |
| Client ID | It is an unique identifier provided to each of the applications while registering in the providers. |
| Client Secret | It is a secret key that is used to authorize the applications. |
| Identifier | It is the property name that holds the email address of the user in the deserialized ID token. |

![Active Directory Settings](/static/assets/on-premise/images/settings/openid-connect-authentication.png)

Active Directory Synchronization Schedule

This section explains how to schedule the synchronization of users and groups from Active Directory with the users and groups in the Bold Reports On-Premise.

Before scheduling the synchronization of Active Directory users and groups, follow the given steps:

1. Configure the [Active Directory Settings](#), before scheduling the synchronization of Active Directory users and groups.
2. To synchronize the Active Directory users, you should import users from the Active Directory to the Bold Reports On-Premise by using the [Import Active Directory Users](#)

3. To synchronize the Active Directory groups, you should import groups from the Active Directory to the Bold Reports On-Premise by using the [Import Active Directory Groups](#)

![Active Directory Synchronization Schedule](/static/assets/on-premise/images/settings/active-directory-schedule-synchronization.png)

Users and groups from Active Directory can be synchronized on schedule to get the latest details of them into the Report Server.

Please find more details on the synchronization of users and groups from Active Directory with Report Server in the below links.

[Synchronize Active Directory Users](#)

[Synchronize Active Directory Groups](#)

Email Notifications

Once a scheduled recurrence of synchronization is completed, the users in the **System Administrator** group will be notified through email about the synchronization status.

Enable/Disable Synchronization schedule

Synchronization schedule can be enabled or disabled anytime from the top check box.

Active Directory Synchronization Schedule

This section explains how to schedule the synchronization of users and groups from Active Directory with the users and groups in the Bold Reports On-Premise.

Before scheduling the synchronization of Active Directory users and groups, follow the given steps:

1. Configure the [Active Directory Settings](#), before scheduling the synchronization of Active Directory users and groups.

2. To synchronize the Active Directory users, you should import users from the Active Directory to the Bold Reports On-Premise by using the [Import Active Directory Users](#)

3. To synchronize the Active Directory groups, you should import groups from the Active Directory to the Bold Reports On-Premise by using the [Import Active Directory Groups](#)

![Active Directory Synchronization Schedule](/static/assets/on-premise/images/settings/active-directory-schedule-synchronization.png)

Users and groups from Active Directory can be synchronized on schedule to get the latest details of them into the Report Server.

Please find more details on the synchronization of users and groups from Active Directory with Report Server in the below links.

[Synchronize Active Directory Users](#)

[Synchronize Active Directory Groups](#)

Email Notifications

Once a scheduled recurrence of synchronization is completed, the users in the **System Administrator** group will be notified through email about the synchronization status.

Enable/Disable Synchronization schedule

Synchronization schedule can be enabled or disabled anytime from the top check box.

Azure Active Directory Synchronization Schedule

This section explains how to schedule the synchronization of users and groups from Azure Active Directory with the users and groups in the Bold Reports On-Premise.

Before you schedule synchronization of Azure Active Directory users and groups, please follow the below steps,

1. Configure [Azure Active Directory Settings](#)
2. To synchronize Azure Active Directory users, you should import users from the Azure Active Directory to Bold Reports On-Premise by referring the following link [Import Azure Active Directory Users](#).
3. To synchronize Azure Active Directory groups, you should import groups from the Azure Active Directory to Bold Reports On-Premise by referring the following link [Import Azure Active Directory Groups](#).

![Active Directory Synchronization Schedule](/static/assets/on-premise/images/settings/azure-active-directory-schedule-synchronization.png)

Users and groups from Azure Active Directory can be synchronized on schedule to get the latest details of them into the Report Server.

Please find more details on the synchronization of users and groups from Azure Active Directory with Report Server in the below links.

[Synchronize Azure Active Directory Users](#)

[Synchronize Azure Active Directory Groups](#)

Email Notifications

Once a scheduled recurrence of synchronization is completed, the users in the **System Administrator** group will be notified through email about the synchronization status.

Enable/Disable Synchronization schedule

Synchronization schedule can be enabled or disabled anytime from the top check box.

Imported Existing Database Users Schedule Synchronization

This section explains how to schedule the synchronization of users from Existing Database with the users in the Bold Reports On-Premise.

Please configure [Database Settings](#), before you schedule synchronization of Existing Database users.

To synchronize existing database users, you should import users from database to Bold Reports On-Premise by referring the following link [Import Database Users](#).

![Imported Database Users Synchronization Schedule](/static/assets/on-premise/images/settings/import-database-user-schedule-synchronization.png)

Users from the Existing Database can be synchronized on schedule to get the latest details of them into the Report Server.

Please find more details on the synchronization of users from Existing Database with Report Server in the below links.

[Synchronize Imported Database Users](#)

Email Notifications

Once a scheduled recurrence of synchronization is completed, the users in the **System Administrator** group will be notified through email about the synchronization status.

Enable/Disable Synchronization schedule

Synchronization schedule can be enabled or disabled anytime from the top check box.

Create Report

This section explains on how to create report and design a report in the Bold Reports On-Premise.

- If the user has **All Reports** permission, then the user can create reports in any category.
- If the user has **Reports in Category** permission with some chosen categories, then the user can only create reports in those chosen categories.
- Reports must be added in any one category.
- Reports can be designed in the Report Designer and then it can be published into the Report Server.
- Reports created by using the SSRS Report Builder can also be uploaded into the Report Server.

Steps to create a report

1. Click on the **[+]** icon from the toolbar and click on the **Create** option in the menu and select **Report**.

![Add button dropdown toggle](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-button-report-server.png)

2. Now, the report designer page opens in a new tab, with a blank report by default.

![Report designer initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/report-designer-initial-view.png)

Read Write Delete permission for that **Specific Report** will be added for the user who created the report.

Create Data

1. To add a data, open the **Data** panel by clicking on the **Data** icon in the configuration panel.

![Configuration panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-icon-configuration-panel.png)

2. Click on **EXPLORE SAMPLE DATA** in the data panel.

![Data creation panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/open-data-panel.png)

3. From the available data, select Sales data and click Add.

![Data creation panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/available-data-list.png)

4. Now, a new DataSource and DataSet will be added in the report.

![Data list view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-list-view.png)

Add a chart report item

The left pane in the design view consists of basic items, data region, data visualization, and sub reports to design an interactive report.

Here, the Chart report item is used for demonstration.

1. Select any of the Chart type in the left pane, then drag and drop it to the design area.

Here, Column chart is used for demonstration.

![Chart report item](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/drag-drop-chart-item.png)

2. The above action will render the Chart report item in the design area.

![Chart initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-initial-view.png)

Assign Data

This step is applicable only for the report items that belongs to data visualization and data region category.

1. To bind data to a report item placed in the design area, focus on that report item.

![Focus chart item](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/focus-report-item.png)

2. Click Properties button in the configuration panel, the property pane opens. Now, switch to DATA tab.

![Chart properties pane](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-properties-pane.png)

3. The DATA tab holds data configuration view.

4. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-assign-drop-down.png)

5. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/measures-dimensions-category.png)

6. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/y-value-chart-design-view.png)

7. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/aggregation-settings-menu.png)

8. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/column-field-design-preview.png)

9. Grouping:

You can group the added column element with another column, by adding the respective dimension element into Row(s) section.

![Achieve grouping by row values](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/row-field-design-preview.png)

Customize the appearance

Navigate to the **PROPERTIES** tab in the properties pane.

![Properties pane](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-properties-tab.png)

This pane holds some general settings and some specific to the report item. Configure the desired settings to the chart for better report design and to improve report readability.

![Chart final design view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/final-view-chart-design.png)

Save report

Once you are done with the report designing, **Save** the report by clicking the **Save** button in the toolbar.

![Save menu](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/designer-save-menu.png)

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/report-preview-page.png)

See also

[Design a Basic Table Report](#)

[Create an Embedded DataSource](#)

[Create an Embedded DataSet](#)

[Link a Shared DataSource into a Report](#)

[Link a Shared DataSet into a Report](#)

[Create a Duplicate Copy of DataSource in a Report](#)

[Create a Duplicate Copy of DataSet in a Report](#)

[Add a Report Parameter to a Report](#)[Embed an Image in a Report](#)

REST API Reference

The following table illustrates the list of available APIs related to Add reports in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|-----------|-------------|--------------------------------------|-------------------------------------------------------------------|
| AddReport | POST | /api/site/{tenant-name}/v1.0/reports | Add report to the server. Report details must be passed as input. |

Create Report

This section explains on how to create report and design a report in the Bold Reports On-Premise.

- If the user has All Reports permission, then the user can create reports in any category.
- If the user has Reports in Category permission with some chosen categories, then the user can only create reports in those chosen categories.
- Reports must be added in any one category.
- Reports can be designed in the Report Designer and then it can be published into the Report Server.
- Reports created by using the SSRS Report Builder can also be uploaded into the Report Server.

Steps to create a report

1. Click on the **[+]** icon from the toolbar and click on the Start from Scratch option.

![Add button dropdown toggle](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-button-report-server.png)

2. Enter the report name in the dialog box and click on

Add and Design button.

![Report designer initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-and-design-button.png)

Report designing in designer will save as draft report.

![Draft report](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/draft-report.png)

3. Now, the report designer page opens in a new tab, with a blank report by default.

![Report designer initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/report-designer-initial-view.png)

Read Write Delete permission for that Specific Report will be added for the user who created the report.

Create Data

1. To add a data, open the Data panel by clicking on the Data icon in the configuration panel.

![Configuration panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-icon-configuration-panel.png)

2. Click on EXPLORE SAMPLE DATA in the data panel.

![Data creation panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/open-data-panel.png)

3. From the available data, select Sales data and click Add.

![Data creation panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/available-data-list.png)

4. Now, a new DataSource and DataSet will be added in the report.

![Data list view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-list-view.png)

Add a chart report item

The left pane in the design view consists of basic items, data region, data visualization, and sub reports to design an interactive report.

Here, the Chart report item is used for demonstration.

1. Select any of the Chart type in the left pane, then drag and drop it to the design area.

Here, Column chart is used for demonstration.

![Chart report item](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/drag-drop-chart-item.png)

2. The above action will render the Chart report item in the design area.

![Chart initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-initial-view.png)

Assign Data

This step is applicable only for the report items that belongs to data visualization and data region category.

1. To bind data to a report item placed in the design area, focus on that report item.

![Focus chart item](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/focus-report-item.png)

2. Click **Properties** button in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-properties-pane.png)

3. The **DATA** tab holds data configuration view.
4. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-assign-drop-down.png)

5. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/measures-dimensions-category.png)

6. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/y-value-chart-design-view.png)

7. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/aggregation-settings-menu.png)

8. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/column-field-design-preview.png)

9. Grouping:

You can group the added column element with another column, by adding the respective dimension element into Row(s) section.

![Achieve grouping by row values](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/row-field-design-preview.png)

Customize the appearance

Navigate to the **PROPERTIES** tab in the properties pane.

![Properties pane](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-properties-tab.png)

This pane holds some general settings and some specific to the report item. Configure the desired settings to the chart for better report design and to improve report readability.

![Chart final design view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/final-view-chart-design.png)

Publish report

Once you are done with the report designing, click on the **Publish** option.

![Save menu](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/designer-publish-option.png)

Preview report

1. To see the report preview, click on the **Preview** button in the center of the report header.

![Preview icon in design view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/report-preview-page.png)

See also

[Design a Basic Table Report](#)

[Create an Embedded DataSource](#)

[Create an Embedded DataSet](#)

[Link a Shared DataSource into a Report](#)

[Link a Shared DataSet into a Report](#)

[Create a Duplicate Copy of DataSource in a Report](#)

[Create a Duplicate Copy of DataSet in a Report](#)

[Add a Report Parameter to a Report](#)

[Embed an Image in a Report](#)

REST API Reference

The following table illustrates the list of available APIs related to Add reports in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|-----------|-------------|--------------------------------------|-------------------------------------------------------------------|
| AddReport | POST | /api/site/{tenant-name}/v1.0/reports | Add report to the server. Report details must be passed as input. |

Create Report

This section explains on how to create report and design a report in the Bold Reports On-Premise.

- If the user has **All Reports** permission, then the user can create reports in any category.
- If the user has **Reports in Category** permission with some chosen categories, then the user can only create reports in those chosen categories.
- Reports must be added in any one category.
- Reports can be designed in the Report Designer and then it can be published into the Report Server.
- Reports created by using the SSRS Report Builder can also be uploaded into the Report Server.

Steps to create a report

1. Click on the **[+]** icon from the toolbar and click on the Start from Scratch option.

![Add button dropdown toggle](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-button-report-server.png)

2. Enter the report name in the dialog box and click on

Add and Design button.

![Report designer initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-and-design-button.png)

Report designing in designer will save as draft report.

![Draft report](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/draft-report.png)

3. Now, the report designer page opens in a new tab, with a blank report by default.

![Report designer initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/report-designer-initial-view.png)

Read Write Delete permission for that Specific Report will be added for the user who created the report.

Create Data

1. To add a data, open the Data panel by clicking on the Data icon in the configuration panel.

![Configuration panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-icon-configuration-panel.png)

2. Click on EXPLORE SAMPLE DATA in the data panel.

![Data creation panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/open-data-panel.png)

3. From the available data, select Sales data and click Add.

![Data creation panel](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/available-data-list.png)

4. Now, a new DataSource and DataSet will be added in the report.

![Data list view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-list-view.png)

Add a chart report item

The left pane in the design view consists of basic items, data region, data visualization, and sub reports to design an interactive report.

Here, the Chart report item is used for demonstration.

1. Select any of the Chart type in the left pane, then drag and drop it to the design area.

Here, Column chart is used for demonstration.

![Chart report item](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/drag-drop-chart-item.png)

2. The above action will render the **Chart** report item in the design area.

![Chart initial view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-initial-view.png)

Assign Data

This step is applicable only for the report items that belongs to **data visualization** and **data region** category.

1. To bind data to a report item placed in the design area, focus on that report item.

![Focus chart item](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/focus-report-item.png)

2. Click **Properties** button in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-properties-pane.png)

3. The **DATA** tab holds data configuration view.
4. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/data-assign-drop-down.png)

5. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/measures-dimensions-category.png)

6. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/y-value-chart-design-view.png)

7. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/aggregation-settings-menu.png)

8. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/column-field-design-preview.png)

9. Grouping:

You can group the added column element with another column, by adding the respective dimension element into **Row(s)** section.

![Achieve grouping by row values](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/row-field-design-preview.png)

[Customize the appearance](#)

Navigate to the **PROPERTIES** tab in the properties pane.

![Properties pane](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/chart-properties-tab.png)

This pane holds some general settings and some specific to the report item. Configure the desired settings to the chart for better report design and to improve report readability.

![Chart final design view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/final-view-chart-design.png)

[Publish report](#)

Once you are done with the report designing, click on the **Publish** option.

![Save menu](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/designer-publish-option.png)

Preview report

1. To see the report preview, click on the **Preview** button in the center of the report header.

![Preview icon in design view](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/manage-content/manage-reports/create-reports/report-preview-page.png)

See also

[Design a Basic Table Report](#)

[Create an Embedded DataSource](#)

[Create an Embedded DataSet](#)

[Link a Shared DataSource into a Report](#)

[Link a Shared DataSet into a Report](#)

[Create a Duplicate Copy of DataSource in a Report](#)

[Create a Duplicate Copy of DataSet in a Report](#)

[Add a Report Parameter to a Report](#)

[Embed an Image in a Report](#)

REST API Reference

The following table illustrates the list of available APIs related to Add reports in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|-----------|-------------|--------------------------------------|-------------------------------------------------------------------|
| AddReport | POST | /api/site/{tenant-name}/v1.0/reports | Add report to the server. Report details must be passed as input. |

Edit report in Report Designer

This section explains on how to edit report in the Report Designer directly from the Bold Reports On-Premise. Reports can be launched directly to the Report Designer from the Report Server.

Click the **Actions** button in the Reports grid context menu and hover over **Edit** to open the Report in web Report Designer.

![Edit report in Report Designer](/static/assets/on-premise/images/manage-content/manage-reports/open-in-Report-designer.png)

Upload report or add report to Report Server

This section explains about how to upload report or add report to Bold Reports On-Premise. Allows you to select the data source and data sets of the report.

Steps to upload a report

1. Click on the  icon from the left side panel and click on the **Upload Report** option.
![Upload button](/static/assets/on-premise/images/manage-content/manage-reports/upload-report-new.png)
2. Select a category for the report and fill in the name and description of the report and upload the RDL file in the Upload Report dialog box.
![Upload Report](/static/assets/on-premise/images/manage-content/manage-reports/upload-report-dialog.png)
3. Click **Save** button.

Update report in the Report Server

This section explains about how to update report using API or through update option from context menu in the Bold Reports On-Premise.

Steps to update a Report

Reports can be updated to move the Report to a different category. Name, description and the Report file(.rdl) can be changed for the Report in the update Report dialog box.

1. Click on the **Update** option in the context menu of the Report to be updated to open the update Report dialog box.
![Update option in context menu](/static/assets/on-premise/images/manage-content/manage-reports/update-context-report.png)
2. Click on the **Update** button in the **Update Report** dialog box after making changes to the Category, Name, Description or to the Report file(.rdl). Comments can also be added if there is a change in the Report file(.rdl) to maintain as **Version Comments**.
![Update Report](/static/assets/on-premise/images/manage-content/manage-reports/update-report.png)

If the report uses shared data sources, then the data sources also should to be selected.

REST API Reference

The following table illustrates the list of available APIs related to update reports in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|--------|-------------|----------|-------------|
|--------|-------------|----------|-------------|

| | | | |
|---------------------|-----|--------------------------------------|-------------------------------------------------------------------------|
| <u>UpdateReport</u> | PUT | /api/site/{tenant-name}/v1.0/reports | Update report to the server. Report details must be passed as input. |
|---------------------|-----|--------------------------------------|-------------------------------------------------------------------------|

Mark favorite report in Report server

This section explains about how to mark favorite report, remove a particular Report from favorites and view the list of favorite Reports in the Bold Reports On-Premise.

Mark a Report as favorite

Reports can be marked as favorite to view them in the **Favorite Reports** category instead of searching them in the Categories or using keywords in the Reports list.

To mark a Report as favorite, click on the star icon near the Report name.

The star icon will be filled with color to indicate that it is added as favorite Report.

![Favorite Report](/static/assets/on-premise/images/manage-content/manage-reports/mark-favorite.png)

Remove a Report from favorites

To remove a Report from favorites, click on the star icon near the Report name. The star icon color will be emptied to indicate that it is removed from favorites.

![Remove Favorite Report](/static/assets/on-premise/images/manage-content/manage-reports/remove-favorite.png)

Favorite Reports Category

Reports that are marked as favorite can be viewed under **Favorite Reports** category.

![Favorite Category](/static/assets/on-premise/images/manage-content/manage-reports/favorite-category.png)

Move, copy and clone reports in Report Server

This section explains on how to move, copy and clone Reports in the Report Designer directly from the Bold Reports On-Premise. Reports can be moved, copied or cloned from one category to another category in the Bold Reports On-Premise.

Move Report

1. Click the **Actions** button in the Reports grid context menu and click **Move** option.

![Move Report in context menu](/static/assets/on-premise/images/manage-content/manage-reports/move-report-option.png)

2. Moves the Report from one to another category.

![Move Reports](/static/assets/on-premise/images/manage-content/manage-reports/move-report.png)

Copy Report

1. Click the **Actions** button in the Reports grid context menu and click **Copy** option.

![Move Report in context menu](/static/assets/on-premise/images/manage-content/manage-reports/copy-report-option.png)

2. Copies the Report from one to another category.

![Copy Reports](/static/assets/on-premise/images/manage-content/manage-reports/copy-report.png)

3. Copies the Report from one Tenant to another Tenant by selecting the Tenant identifier.

![Copy Report to Tenant](/static/assets/on-premise/images/manage-content/manage-reports/copy-report-to-tenant.png)

Clone Reports

1. Click the **Actions** button in the Reports grid context menu and click **Clone** option.

![Move Report in context menu](/static/assets/on-premise/images/manage-content/manage-reports/clone-report-option.png)

2. Creates a reference of the Report to destination category. When the Report .rdl file is changed, then it affects the Reports in both the categories.

![Clone Reports](/static/assets/on-premise/images/manage-content/manage-reports/clone-report.png)

Share report with users in the Report Server

This section explains about how to share report with users by adding permission in the Bold Reports On-Premise.

Steps to share a Report

1. Click the **Share Permissions** icon in the Reports grid context menu.

![Manage Permission](/static/assets/on-premise/images/manage-content/manage-reports/manage-permission-context.png)

2. Click the **Manage Access** button.

![Manage Access Button](/static/assets/on-premise/images/manage-content/manage-reports/manage-access-button.png)

3. Select the permission access and the users or groups to share the dashboard.

![Share Report](/static/assets/on-premise/images/manage-content/manage-reports/share-report.png)

4. After selecting the access and users or groups, click on the **Add** button.

![Add button](/static/assets/on-premise/images/manage-content/manage-reports/add-button.png)

Only the user who created the Report can share the Report with other Report Server users.

Remove Permission

The user who created the report and the Administrator can remove the shared report permissions using the **Remove** icon in the **Actions** column of the each permissions.

![Add Permission](/static/assets/on-premise/images/manage-content/manage-reports/remove-permission.png)

share report link to the users

This section explains on how to share report link to the users from the Bold Reports On-Premise reports.

If the Report is public, anyone with this link will be able to view its contents.

If the Report is private, anyone with this link can navigate to the Report, but only users with the appropriate permissions will be able to view its contents.

Get Link

Get Link option is available for all the Reports.

Follow the steps below to get the Reports link.

1. Click on the context menu of the respective Report and choose **Get Link** option.

![Get Link Menu](/static/assets/on-premise/images/manage-content/manage-reports/get-link-menu.png)

2. Respective Report link will be shown in the **Get Link** dialog box.

For Public Reports

![Get Link Dialog](/static/assets/on-premise/images/manage-content/manage-reports/get-link-public.png)

For Private Reports

![Get Link Dialog](/static/assets/on-premise/images/manage-content/manage-reports/get-link-private.png)

Public report

This section explains on how to make public report to allow access to anonymous users and make private reports to access only to registered users in the Report Server.

Public Reports are accessible to anonymous users who has the Report link.

Private Reports are accessible to the registered users in the Report Server who has appropriate permissions.

Make public

Make Public option is available only to the owner of the Report.

Follow the steps below to make the Reports accessible to anonymous users.

1. Click on the context menu of the respective Report and choose Make Public option.

![Make public menu](/static/assets/on-premise/images/manage-content/manage-reports/make-public-menu.png)

2. Click on Make Public in the following confirmation dialog box.

![Make public Dialog](/static/assets/on-premise/images/manage-content/manage-reports/make-public-dialog.png)

3. Once the Report made public, dialog box with the Report link will be displayed. You can copy and share the link to the users.

![Get link public Dialog](/static/assets/on-premise/images/manage-content/manage-reports/get-link-public.png)

Make Private

Make Private option is available only to the owner of the Report.

Follow the steps below to make the Reports accessible only to the users in the Report Server who has appropriate permissions.

1. Click on the context menu of the respective Report and choose Make Private option.

![Make Private menu](/static/assets/on-premise/images/manage-content/manage-reports/make-private-menu.png)

2. Click on Make Private in the following confirmation dialog box.

![Make Private dialog](/static/assets/on-premise/images/manage-content/manage-reports/make-private-dialog.png)

Once the Report made private dialog box confirmation message will be displayed.

Public Reports

Public reports are listed under the public section as shown in the below image.

![Public Reports](/static/assets/on-premise/images/manage-content/manage-reports/public-reports.png)

Delete report from Report Server

This section explains about how to delete report individually or delete multiple reports from the Bold Reports On-Premise. Reports can also be deleted from the Report Server when they are no longer required.

Click the **Actions** button in the Reports grid context menu and click **Delete** to delete the Report.

![Delete Report](/static/assets/on-premise/images/manage-content/manage-reports/delete-report.png)

Reports cannot be deleted when they are scheduled by an user.

[Delete All Sample Reports](#)

All Sample Reports can be deleted in a simpler way by deleting the Sample Category **Sample Reports**, **Sample Reports (Web Designer)**.

Click the **Actions** icon in the context menu and click **Delete** to delete all sample reports as well as sample category.

![Delete All Sample Reports](/static/assets/on-premise/images/manage-content/manage-reports/delete-all-sample-reports.png)

If the sample category has any modified sample report, or scheduled report, or newly added report, those reports will be skipped and the category will not be deleted.

[REST API Reference](#)

The following table illustrates the list of available APIs related to delete reports in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|------------------------------|-------------|-----------------------------------------|--------------------------------------------------------------------------------------------------|
| DeleteReport | DELETE | /api/site/{tenant-name}/v1.0/items/{id} | Deletes the specified report from the server. Report item ID should be passed in path parameter. |

Version history of report

This section explains on how to view version history of report in the Bold Reports On-Premise. Versions and file logs for each Report will be maintained in the Report Server for every changes in the Report.

Click the **Actions** button in the Reports grid context menu and click **Version History** option.

![Version History Option](/static/assets/on-premise/images/manage-content/manage-reports/versions-history-option.png)

[Versions](#)

For each change in the **.rdl** file, a new version will be created. All versions can be individually opened or edited in designer. At any time, the Report can be rolled back to an older version.

![Versions](/static/assets/on-premise/images/manage-content/manage-reports/versions.png)

[File logs](#)

For each change in the Report including changes in the event, user and date, Report Server logs the changes done in the file logs.

![File logs](/static/assets/on-premise/images/manage-content/manage-reports/file-logs.png)

Download Report from Report Server

This section explains about how to download report from the Bold Reports On-Premise and share with other users.

Click the **Actions** button in the Reports grid context menu and click **Download Report** to download the Report in **.rdl** format.

![Download Report](/static/assets/on-premise/images/manage-content/manage-reports/download-report.png)

This option is available on Bold Reports On-Premise Edition from **2.2.28** version.

Manage the Report Views

Report Views is a saved parameters information for the report and can be applied to viewer while viewing the report with Report Server. This can be shared, updated, and deleted only to the user who created it.

This section explains how to open, add, update, share, and delete Report Views in the Bold Reports On-Premise.

Add the Report Views

1. Click the view icon at top right corner, select **Save** to add a report view.

![View Icon](/static/assets/on-premise/images/manage-content/manage-reports/view-icon.png)

![Save View](/static/assets/on-premise/images/manage-content/manage-reports/save-icon.png)

Enter a name for the new view and click the **Tick** as shown in the following figure.

![Create Report View](/static/assets/on-premise/images/manage-content/manage-reports/add-report-view.png)

After views saved, you can save the same view again in a new name by clicking the **Save As** icon at the top right corner.

![Create Report View](/static/assets/on-premise/images/manage-content/manage-reports/save-as-icon.png)

View icon will be available for the report, which contains parameter.

2. Views will be added and saved Report Views will be displayed in the **Saved Views** panel.

![View Report views](/static/assets/on-premise/images/manage-content/manage-reports/view-saved-report-views.png)

Open the Report Views

Saved Report Views of each report will be listed in the context menu of the respective report.

![Manage Reports view](/static/assets/on-premise/images/manage-content/manage-reports/report-views-from-server.png)

Share the Report Views

Saved Report Views can be shared with other users and groups in the Bold Report Server. They can be also shared to anonymous users.

To share the Report Views, click the  icon of the respective report view from **Saved Views** panel.

![Share Report View](/static/assets/on-premise/images/manage-content/manage-reports/share-icon.png)

Choose the users and groups from the drop-down and click **Share** to share to the selected users and groups.

![Share Report View popup](/static/assets/on-premise/images/manage-content/manage-reports/share-report-view.png)

Copy the Report Views link

To copy the Report Views link, click the  icon of the respective report view from **Saved Views** panel.

![Delete Report](/static/assets/on-premise/images/manage-content/manage-reports/copy-view.png)

Delete the Report Views

To delete the Report Views, click the  icon of the respective report view from **Saved Views** panel.

![Delete Report](/static/assets/on-premise/images/manage-content/manage-reports/delete-report-view.png)

Data sources management in Report Server

You can create and store a data source on the Report Server when you have data source that you use often. When you create or upload a data source to Report Server, it is considered as shared data source that can be used by multiple reports.

It is recommended to use shared data sources as much as possible. It is easier to manage, and help to keep reports and the data sources access more secure. The shared data sources always placed in the **Data Sources** page, they are accessible to the user depending on the user's permission.

This section about data sources management in Report Server. Allows you to add,delete shared data sources in the Report Server. The following image shows the list of options available to manage a datasources.

![Options available to manage data sources in the Report Server](/static/assets/on-premise/images/manage-content/manage-data-sources/managing-data-sources.png)

Add or create data sources

Data source can be added to Report Server only if the user has **Create All Data Sources** permission.

Steps to add a data source

1. Click on the **Create Data Source** button from Data Sources listing page.

![Create data source menu option](/static/assets/on-premise/images/manage-content/manage-data-sources/create-data-source-menu-option.png)

While adding datasource we can also add new Dataset to the Report Server either by using **Create New Datasource** option or **Use existing Datasource** option.

2. By default, **Create New Datasource** will be chosen. Enter the name and description of the data source, the data source type, connection string and the connection detail to connect the selected data source type. Data source can be created with connections to any one of the following data source types,

![Set data source properties](/static/assets/on-premise/images/manage-content/manage-data-sources/add-data-source.png)

- SQL
- SQLCE
- OLEDB
- ODBC
- Oracle
- XML
- SSAS
- PostgreSQL

![Set data source existing](/static/assets/on-premise/images/manage-content/manage-data-sources/add-data-source-existing.png)

3. Click on **Connect** button, it will add new dataset by connecting with newly created or existing datasource.
4. Enter the name and description of the dataset and click on **Proceed to Designer** option.

![Proceed to designer](/static/assets/on-premise/images/manage-content/manage-data-sources/proceed-to-designer.png).

5. After designing dataset click on the **save** option.

Read Write Delete Download permission for the **Specific Data Source** will be added for the user who created the data source.

Share data sources

This section explains on how to share data sources with the other users in the Report Server.

Steps to share a data source

1. Click the **Actions** button in the Data sources grid context menu and select **Sharing Permissions** option.

![Sharing permission menu option](/static/assets/on-premise/images/manage-content/manage-data-sources/manage-datasource-permission-context-menu.png)

2. Sharing permission dialog will open. By default it will open dialog with Share To tab. Select the permission access from the Select Access dropdown and select the users or groups to share the data source.

![Select data source access permission](/static/assets/on-premise/images/manage-content/manage-data-sources/select-datasource-permission-dialog.png)

3. After selecting the access and users or groups, click on the Share button.

![Add permission to a data source](/static/assets/on-premise/images/manage-content/manage-data-sources/add-datasource-permission.png)

Only the user who created the data source can share the data source with other Report Server users.

Update data sources

This section explains how to update the data sources in Bold Reports.

Steps to update the data source

1. Click the Actions in the data sources grid context menu and select Update option.

![Update datasource menu option](/static/assets/on-premise/images/manage-content/manage-data-sources/update-datasource-option.png)

2. The name, description, data source type, connection string, and credential details to connect to the specified data source type can be changed by clicking Update option in the update data source dialog box.

![Update option](/static/assets/on-premise/images/manage-content/manage-data-sources/update-option.png)

This option is available on Bold Reports On-Premise Edition from 2.2.28 version.

Remove data source permission

Click on the Shared With tab. The user who created the data source can remove the shared data source permissions using the Remove option in the Actions column of the each permissions.

![Remove permission of a data source](/static/assets/on-premise/images/manage-content/manage-data-sources/remove-datasource-permission.png)

Delete data source

Data sources can also be deleted from the Report Server when they are no longer required.

Click the Actions button in the data sources grid context menu and select Delete to delete the data source.

![Delete a data source](/static/assets/on-premise/images/manage-content/manage-data-sources/delete-data-source.png)

Data Sources cannot be deleted if a report uses it.

REST API Reference

The following table illustrates the list of available APIs related to datasources in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|-----------------------------------------------|-------------|------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>AddDataSource</u> | POST | /api/site/{tenant-name}/v1.0/reports/data-sources | Adds a new datasource to the server. Datasource details must be passed as input. |
| <u>UpdateDataSource</u> | PUT | /api/site/{tenant-name}/v1.0/reports/data-sources | Updates the datasource in the server. Updated datasource details must be passed as input. |
| <u>GetDatasources</u> | GET | /api/site/{tenant-name}/v1.0/items | Returns the list of datasources for current user. ItemType should be Datasource. |
| <u>GetDataSourceLocation</u> | GET | /api/site/{tenant-name}/v1.0/items/{itemId}/location | Returns the item location of the specified datasource. Specific DataSource ID should be passed in path parameter. |
| <u>IsDataSourceNameExists</u> | POST | /api/site/{tenant-name}/v1.0/items/is-name-exists | Returns an item existence whether the given datasource name already exists or not in server. Datasource name and ItemType as DataSource should be passed in request body. |
| <u>GetDataSourceDetail</u> | GET | /api/site/{tenant-name}/v1.0/items/{id} | Returns the specified datasource details from the server. DataSource item ID should be passed in path parameter. |
| <u>DeleteDataSource</u> | DELETE | /api/site/{tenant-name}/v1.0/items/{id} | Deletes the specified datasource from the server. DataSource item ID should be passed in path parameter. |

Dataset management in Report Server

You can create and store dataset on a Report Server that can be used by multiple reports. When you create or upload a dataset to Report Server, it is considered as shared dataset and it always placed in the **Datasets** page. A shared dataset must be based on a shared data source. Datasets are accessible to the user depending on the user's permission. This section explains about dataset management in Report Server. Allows you to create, edit, update, sharing permissions and delete data sets in the Bold Reports On-Premise. The following image shows the list of options available to manage dataset.

![Options available to manage data sets](/static/assets/on-premise/images/manage-content/manage-data-sets/managing-datasets.png)

Create dataset

Dataset can be created only if the user has **Create All Datasets** permission.

Steps to create a dataset

We can create dataset by following ways.

1. Add dataset from + icon.
2. Add dataset from Dataset listing page.

Add dataset from `+` icon

- Click on the + button in the left side menu and choose **Create Dataset** to add a dataset.

![Create a dataset menu option](/static/assets/on-premise/images/manage-content/manage-data-sets/create-dataset-option.png)

Add dataset from Dataset listing page

- Click **Dataset** from left side panel of the Report Server and click **Create my first Dataset** button there is no dataset in listing page or else click **Create Dataset** button.

![Create a dataset menu option listing](/static/assets/on-premise/images/manage-content/manage-data-sets/create-dataset-option-listing.png)

- To create dataset, we have use datasource either newly created or else existing datasource.
Refer this link for [creating dataset](#)

Read Write Delete permission for that **Specific Dataset** will be added for the user who created the dataset.

Add or upload dataset

1. Click on the + button in the left side menu and choose **Upload Dataset** and select **Dataset** to add a dataset.

![Upload DataSet](/static/assets/on-premise/images/manage-content/manage-data-sets/upload-dataset-option.png)

2. Enter the name and description of the dataset.
3. Click **Browse** and select the **.rsd** file to upload.
4. Select the **DataSource**. In the select data source dialog, you can either select a data source that is already available in the Report Server or create a new data source at that time itself.

![Select a data source for the uploaded dataset](/static/assets/on-premise/images/manage-content/manage-data-sets/upload-dataset-dialog.png)

5. Click on the **Save** option, the dataset will be uploaded into the Bold Reports On-Premise and displayed in the list as shown below,

![Added datasets list view](/static/assets/on-premise/images/manage-content/manage-data-sets/uploaded-datasets-list.png)

Read Write Delete permission for that **Specific Dataset** will be added for the user who created the dataset.

Create report with dataset

Click the **Actions** button in the dataset grid context menu and select **Create Reports** to create a new report with a dataset.

![Create new report with dataset](/static/assets/on-premise/images/manage-content/manage-data-sets/create-new-report-with-dataset.png)

Edit dataset

Click the **Actions** button in the dataset grid context menu and select **Edit Dataset** to edit a dataset.

![Edit dataset properties](/static/assets/on-premise/images/manage-content/manage-data-sets/edit-dataset.png)

Share dataset

This section explains on how to share dataset with the other users in the Report Server.

Steps to share a dataset

1. Click the **Actions** button in the dataset grid context menu and select **Sharing Permissions** option.

![Manage dataset permission option](/static/assets/on-premise/images/manage-content/manage-data-sets/manage-dataset-permissions.png)

2. Sharing permission dialog will open. By default it will open dialog with **Share To** tab. Select the permission access from the **Select Access** dropdown and select the users or groups to share the dataset.

![Set permissions to share a dataset](/static/assets/on-premise/images/manage-content/manage-data-sets/manage-dataset-permission-dialog.png)

3. After selecting the access and users or groups, click on the **Add Permission** button.

![Add permission to dataset](/static/assets/on-premise/images/manage-content/manage-data-sets/add-dataset-permission.png)

Only the user who created the dataset can share the dataset with other Report Server users.

Remove permission

Click on the Shared With tab. The user who created the dataset can remove the shared dataset permissions using the Remove option in the Actions column of the each permissions.

![Remove Permission](/static/assets/on-premise/images/manage-content/manage-data-sets/remove-dataset-permission.png)

Version history

Versions and file logs for each dataset are maintained in the Report Server for every changes in the Dataset. Click the Actions button in the grid context menu and click Version History option.

![Version History Option](/static/assets/on-premise/images/manage-content/manage-data-sets/versions-history-option.png)

Versions

For each change in the .rsd file, a new version will be created. All versions can be individually opened.

![Versions](/static/assets/on-premise/images/manage-content/manage-data-sets/versions.png)

File logs

For each change in the dataset including changes in the event, user and date.

![File logs](/static/assets/on-premise/images/manage-content/manage-data-sets/file-logs.png)

Update dataset

1. Click the Actions button in the dataset grid context menu and select Update Dataset to update the Dataset.

![Update DataSet](/static/assets/on-premise/images/manage-content/manage-data-sets/update-dataset-option.png)

2. Enter the name and description of the dataset.
3. Click Browse and select the .rsd file to upload.
4. Select the DataSource. In the select data source dialog, you can either select a data source that is already available in the Report Server or create a new data source at that time itself.

![Select a data source for the uploaded dataset](/static/assets/on-premise/images/manage-content/manage-data-sets/update-dataset-dialog.png)

5. Click on the Update option, the dataset will be uploaded into the Bold Reports On-Premise and displayed in the list as shown below,

![Added datasets list view](/static/assets/on-premise/images/manage-content/manage-data-sets/uploaded-datasets-list.png)

Delete dataset

Datasets can also be deleted from the Report Server when they are no longer required.

Click the **Actions** button in the dataset grid context menu and select **Delete** to delete the Dataset.

![Delete Dataset](/static/assets/on-premise/images/manage-content/manage-data-sets/delete-datasets.png)

Dataset cannot be deleted if any report uses it.

REST API Reference

The following table illustrates the list of available APIs related to datasets in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|--------------------------------------------|-------------|------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>AddDataset</u> | POST | /api/site/{tenant-name}/v1.0/reports/datasets | Adds a new dataset to the server. Dataset details must be passed as input. |
| <u>UpdateDataset</u> | PUT | /api/site/{tenant-name}/v1.0/reports/datasets | Updates the dataset in the server. Updated dataset details must be passed as input. |
| <u>GetDataSets</u> | GET | /api/site/{tenant-name}/v1.0/items | Returns the list of datasets for current user. ItemType should be Dataset. |
| <u>GetDataSetLocation</u> | GET | /api/site/{tenant-name}/v1.0/items/{itemId}/location | Returns the item location of the specified dataset. Specific DataSet ID should be passed in path parameter. |
| <u>IsDataSetNameExists</u> | POST | /api/site/{tenant-name}/v1.0/items/is-name-exists | Returns an item existence whether the given dataset name already exists or not in server. Dataset name and ItemType as DataSet should be passed in request body. |
| <u>GetDataSetDetail</u> | GET | /api/site/{tenant-name}/v1.0/items/{id} | Returns the specified dataset details from the server. DataSet item ID should be passed in path parameter. |
| <u>DeleteDataSet</u> | DELETE | /api/site/{tenant-name}/v1.0/items/{id} | Deletes the specified dataset from the server. DataSet item ID should be passed in path parameter. |

Manage report schedules

Reporting Server provides report-specific schedules to help you control processing and distribution of reports. All schedules specify a type of recurrence: monthly, weekly, or daily.

This section explains on how to add, edit, delete schedules, manage report schedules and also on how to run the schedules on demand and enable or disable schedules in the Bold Reports On-Premise.

Schedules page displays the schedules that are accessible by the user depending on the user's permission will be shown in the schedules page.

![Manage Schedules](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/add-schedule-1.png)

Add schedule

Schedules can be created only if the user has **Create All Schedules** permission. Schedules can be created in two ways,

1. Add schedule from + button in the left side menu.
2. Add schedule from context menu of the respective reports.
3. Add schedule from schedule listing page.

Add schedule from `+` button menu

- Click on the + button in the left side menu and choose **Schedule** under **Create** to add a Schedule.

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/create-schedule1.png)

- Select the required category from **Category** dropdown. After selecting the category, corresponding reports under that selected category will be displayed in the **report** dropdown,

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/select-category.png)

- Select the required report from the dropdown.

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/select-report.png)

Add schedule from context menu of the respective reports

- Click the **Actions** button in the reports grid context menu and select **Schedule** to schedule the corresponding report.

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/create-schedule.png)

- Once dialog was opened, the category and report values are selected by default,

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/category-report-preselect.png)

Categories or Reports can be changed from schedule dialog box itself.

Add schedule from [schedule listing page](#)

- Click **Schedules** from left side panel of the Report Server and click **Create Schedule** button.

![Create Schedule from schedule page](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/add-schedule-from-schedule-page.png)

- Parameters available for the report will be shown in scheduler dialog.

![Set Parameter](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/set-parameter.png)

- Select the parameter values to schedule and export the reports with this filter.

![Available Parameter](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/available-parameters.png)

To modify the parameter values, you should enable the **Set parameters** option in the schedule dialog box.

- Select the recurrence type, recurrence, start and end dates, export formats and the users to which the exported reports has to be emailed in the 'Add Schedule' dialog box.
- Reports can be scheduled hourly, daily, weekly, monthly and yearly
- Reports can be exported in PDF, Word, Excel, HTML, PPT and CSV formats
- Application time zone will be shown below the date picker. Start time of the schedule will be converted to client time zone and shown in the right side for user's convenience.

![Add Schedule 1](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/add-schedule-1.png)

- You can compress the exporting reports as a zip file by selecting the **Enable File Compression** option in the schedule dialog box. This is not mandatory and you can decompress it at any time by simply unchecking that option.

![File Compression](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/file-compression.png)

- You can secure the exporting reports with password protection by enabling the **Enable Password Protection** option in the schedule dialog box. By default, the **Default Password** option is selected.

For default password, the password will be generated in the combination of first four characters of username and last four characters of email.

- You can customize the password protection rules for exporting reports by choosing the **Custom Password** option.
- But, the username or first name is mandatory in the password condition.

![Custom Password](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/custom-password.png)

For instance, you have a sample user with details below:

First Name : John

Last Name :

Email: johnmichael@gmail.com

With this sample user, the possible password combination and its unlock password have been explained below.

| Password Rule | Unlock Password |
|--------------------------------------------------------------------------------|-----------------|
| First four characters of Email + Last four characters of First Name | MichJohn |
| First four characters of First Name + First four characters of Last Name | John |
| First four characters of Email + Last four characters of Last Name | johnhael |

Mail template customization

- Mail template define the text which is sent via email to the report server user.
- Initially the default template will displayed in the mail template and if the custom password is enable the user can add the password hint in the mail template if necessary
- And also you can customize the email template with your preference.

![Customize Email Template](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/customize-email-template.png)

Mail template predefined variables

By selecting the variable the scheduled user variable is assigned directly from the server to the mail template

- {Full Name}: Full name of the recipient
- {First Name}: First name of the recipient
- {Last Name}: Last name of the recipient
- {Schedule Name}: Name of the schedule
- {Report Name}: Scheduled report name

- {Report Link}: Link to the scheduled report
- {Export Format}: Export file format which is chosen the schedule
- {Organization Name}: [Organization name](#) is retrieved from the site setting
- Exported reports can be sent to individual users or groups by checking **Email attachment** option.

![Email Attachment Checkbox](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/email-attachment-checkbox.png)

- We can also save exported reports into any location by checking **Save as file** option.

![Save As Checkbox](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/save-as-checkbox.png)

- To save the exported report, **Export Path** and **Max reports count** has to be filled.

Export Path - Location to save the exported report. By default, the location will be **C:\Syncfusion\Report Server\ReportServer.Web\App_Data**.

Max reports count - Maximum exported reports count to be save in that location.

- When clicking the **Schedule**, the report will be scheduled in the selected recurrence.

Read Write Delete permission for that **Specific Schedule** will be added for the user who created the schedule.

Edit schedule

Category, report, name, recurrence type, recurrence, start and end dates, export format and the recipients can be changed in the **Edit Schedule** dialog box.

Run now

Schedules can be made to run on demand by using this **Run Now** option in the schedule grid context menu. Report will be exported in the format specified and sent to the recipients.

![Run now schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/run-now-schedule.png)

Enable or disable schedule

Schedules can be disabled at any time which will ignore any next occurrences. When enabled it will get the next occurrence and run accordingly.

![Enable Disable Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/enable-disable-schedule.png)

Delete schedules

Schedules can be deleted from the Report Server when it is no longer required.

Click the **Actions** button in the schedules grid context menu and select **Delete** to delete the schedule.

![Delete schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/delete-schedule.png)

REST API reference

The following table illustrates the list of available APIs related to schedules in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|------------------------------|-------------|------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>AddSchedule</u> | POST | /api/site/{tenant-name}/v1.0/reports/schedule | Adds schedule to the server. Schedule details should be passed as input. |
| <u>UpdateSchedule</u> | PUT | /api/site/{tenant-name}/v1.0/reports/schedule/{scheduleId} | Updates schedule. Should provide Schedule ID in path parameter and schedule details as input. |
| <u>RunScheduleReport</u> | GET | /api/site/{tenant-name}/v1.0/schedules/{scheduleId}/run | Runs scheduled report. Should provide Schedule ID in path parameter. |
| <u>GetScheduleItemDetail</u> | GET | /api/site/{tenant-name}/v1.0/reports/schedule/{scheduleId} | Returns the respective schedule details. Should pass Schedule ID in path parameter. |
| <u>GetScheduleItems</u> | GET | /api/site/{tenant-name}/v1.0/reports/schedule/items | Returns list of scheduled items. |
| <u>GetSchedules</u> | GET | /api/site/{tenant-name}/v1.0/items | Returns the list of schedules for current user. ItemType should be Schedule. |
| <u>IsScheduleNameExists</u> | POST | /api/site/{tenant-name}/v1.0/items/is-name-exists | Returns an item existence whether the given schedule name already exists or not in server. Schedule name and ItemType as Schedule should be passed in request body. |
| <u>GetScheduleDetail</u> | GET | /api/site/{tenant-name}/v1.0/items/{id} | Returns the specified schedule details from the server. Schedule item ID should be passed in path parameter. |
| <u>DeleteSchedule</u> | DELETE | /api/site/{tenant-name}/v1.0/items/{id} | Deletes the specified schedule from the server. Schedule item ID should |

| | | | |
|--|--|--|------------------------------|
| | | | be passed in path parameter. |
|--|--|--|------------------------------|

Manage report schedules

Reporting Server provides report-specific schedules to help you control processing and distribution of reports. All schedules specify a type of recurrence: monthly, weekly, or daily.

This section explains on how to add, edit, delete schedules, manage report schedules and also on how to run the schedules on demand and enable or disable schedules in the Bold Reports On-Premise.

Schedules page displays the schedules that are accessible by the user depending on the user's permission will be shown in the schedules page.

![Manage Schedules](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/add-schedule-1.png)

Add schedule

Schedules can be created only if the user has **Create All Schedules** permission. Schedules can be created in two ways,

1. Add schedule from + button in the left side menu.
2. Add schedule from context menu of the respective reports.
3. Add schedule from schedule listing page.

Add schedule from '+' button menu

- Click on the + button in the left side menu and choose **Schedule** under **Create** to add a Schedule.

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/create-schedule1.png)

- Select the required category from **Category** dropdown. After selecting the category, corresponding reports under that selected category will be displayed in the **report** dropdown,

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/select-category.png)

- Select the required report from the dropdown.

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/select-report.png)

Add schedule from context menu of the respective reports

- Click the **Actions** button in the reports grid context menu and select **Schedule** to schedule the corresponding report.

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/create-schedule.png)

- Once dialog was opened, the category and report values are selected by default,

![Create Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/category-report-preselect.png)

Categories or Reports can be changed from schedule dialog box itself.

Add schedule from schedule listing page

- Click **Schedules** from left side panel of the Report Server and click **Create Schedule** button.

![Create Schedule from schedule page](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/add-schedule-from-schedule-page.png)

- Parameters available for the report will be shown in scheduler dialog.

![Set Parameter](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/set-parameter.png)

- Select the parameter values to schedule and export the reports with this filter.

![Available Parameter](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/available-parameters.png)

To modify the parameter values, you should enable the **Set parameters** option in the schedule dialog box.

- Select the recurrence type, recurrence, start and end dates, export formats and the users to which the exported reports has to be emailed in the 'Add Schedule' dialog box.
- Reports can be scheduled hourly, daily, weekly, monthly and yearly
- Reports can be exported in PDF, Word, Excel, HTML, PPT and CSV formats
- Application time zone will be shown below the date picker. Start time of the schedule will be converted to client time zone and shown in the right side for user's convenience.

![Add Schedule 1](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/add-schedule-1.png)

- You can compress the exporting reports as a zip file by selecting the **Enable File Compression** option in the schedule dialog box. This is not mandatory and you can decompress it at any time by simply unchecking that option.

![File Compression](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/file-compression.png)

- You can secure the exporting reports with password protection by enabling the **Enable Password Protection** option in the schedule dialog box. By default, the **Default Password** option is selected.

For default password, the password will be generated in the combination of first four characters of username and last four characters of email.

- You can customize the password protection rules for exporting reports by choosing the **Custom Password** option.
- But, the username or first name is mandatory in the password condition.

![Custom Password](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/custom-password.png)

For instance, you have a sample user with details below:

First Name : John

Last Name :

Email: johnmichael@gmail.com

With this sample user, the possible password combination and its unlock password have been explained below.

| Password Rule | Unlock Password |
|--------------------------------------------------------------------------------|-----------------|
| First four characters of Email + Last four characters of First Name | MichJohn |
| First four characters of First Name + First four characters of Last Name | John |
| First four characters of Email + Last four characters of Last Name | johnhael |

Mail template customization

- Mail template define the text which is sent via email to the report server user.
- Initially the default template will displayed in the mail template and if the custom password is enable the user can add the password hint in the mail template if necessary
- And also you can customize the email template with your preference.

![Customize Email Template](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/customize-email-template.png)

Mail template predefined variables

By selecting the variable the scheduled user variable is assigned directly from the server to the mail template

- {Full Name}: Full name of the recipient
- {First Name}: First name of the recipient
- {Last Name}: Last name of the recipient
- {Schedule Name}: Name of the schedule
- {Report Name}: Scheduled report name
- {Report Link}: Link to the scheduled report
- {Export Format}: Export file format which is chosen the schedule
- {Organization Name}: [Organization name](#) is retrieved from the site setting
- Exported reports can be sent to individual users or groups by checking **Email attachment** option.

![Email Attachment Checkbox](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/email-attachment-checkbox.png)

- We can also save exported reports into any location by checking **Save as file** option.

![Save As Checkbox](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/save-as-checkbox.png)

- To save the exported report, **Export Path** and **Max reports count** has to be filled.

Export Path - Location to save the exported report. By default, the location will be **C:\Syncfusion\Report Server\ReportServer.Web\App_Data**.

Max reports count - Maximum exported reports count to be save in that location.

- When clicking the **Schedule**, the report will be scheduled in the selected recurrence.

Read Write Delete permission for that **Specific Schedule** will be added for the user who created the schedule.

Edit schedule

Category, report, name, recurrence type, recurrence, start and end dates, export format and the recipients can be changed in the **Edit Schedule** dialog box.

Run now

Schedules can be made to run on demand by using this **Run Now** option in the schedule grid context menu. Report will be exported in the format specified and sent to the recipients.

![Run now schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/run-now-schedule.png)

Enable or disable schedule

Schedules can be disabled at any time which will ignore any next occurrences. When enabled it will get the next occurrence and run accordingly.

Manage report schedules

Delete schedules

![Enable Disable Schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/enable-disable-schedule.png)

Delete schedules

Schedules can be deleted from the Report Server when it is no longer required.

Click the **Actions** button in the schedules grid context menu and select **Delete** to delete the schedule.

![Delete schedule](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/delete-schedule.png)

REST API reference

The following table illustrates the list of available APIs related to schedules in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|-----------------------|-------------|------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AddSchedule | POST | /api/site/{tenant-name}/v1.0/reports/schedule | Adds schedule to the server. Schedule details should be passed as input. |
| UpdateSchedule | PUT | /api/site/{tenant-name}/v1.0/reports/schedule/{scheduleId} | Updates schedule. Should provide Schedule ID in path parameter and schedule details as input. |
| RunScheduleReport | GET | /api/site/{tenant-name}/v1.0/schedules/{scheduleId}/run | Runs scheduled report. Should provide Schedule ID in path parameter. |
| GetScheduleItemDetail | GET | /api/site/{tenant-name}/v1.0/reports/schedule/{scheduleId} | Returns the respective schedule details. Should pass Schedule ID in path parameter. |
| GetScheduleItems | GET | /api/site/{tenant-name}/v1.0/reports/schedule/items | Returns list of scheduled items. |
| GetSchedules | GET | /api/site/{tenant-name}/v1.0/items | Returns the list of schedules for current user. ItemType should be Schedule. |
| IsScheduleNameExists | POST | /api/site/{tenant-name}/v1.0/items/is-name-exists | Returns an item existence whether the given schedule name already exists or not in server. Schedule name and ItemType as Schedule should be passed in request body. |

| | | | |
|--------------------------|--------|-----------------------------------------|--------------------------------------------------------------------------------------------------------------|
| <u>GetScheduleDetail</u> | GET | /api/site/{tenant-name}/v1.0/items/{id} | Returns the specified schedule details from the server. Schedule item ID should be passed in path parameter. |
| <u>DeleteSchedule</u> | DELETE | /api/site/{tenant-name}/v1.0/items/{id} | Deletes the specified schedule from the server. Schedule item ID should be passed in path parameter. |

Schedule report settings

This section explain about schedule report settings to compress the exporting reports as a zip file by using the **Enable File Compression** option.

This is optional setting and you can export the uncompressed report by simply unchecking **Enable File Compression** option.

Enable file compression

1. Click on the **Settings** option in Report Server left-side panel and navigate to **Reports** tab.
2. Switch to **Schedule** tab as in following image

![Schedule report settings tab in Report Server](/static/assets/on-premise/images/manage-schedule/schedule-settings/schedule-report-settings-tab.png)

3. Check the **Enable File Compression** option and click on the **Save** button to save the settings.
4. You can secure the exporting reports with password protection by enabling the **Enable Password Protection** option. By default, the **Default Password** option is selected.

For default password, the password will be generated in the combination of first and last four characters of email.

5. You can customize the password protection rules for exporting reports by choosing the **Custom Password** option. But, the username or first name is mandatory in the password condition.

![Custom password compression settings](/static/assets/on-premise/images/manage-schedule/schedule-settings/custom-password-for-compression.png)

6. For instance, you have a sample user with details below:

First Name : John

Last Name :

Email: johnmichael@gmail.com

7. With this sample user, the possible password combination and its unlock password have been explained below.

| Password Rule | Unlock Password |
|--------------------------------------------------------------------------------|-----------------|
| First four characters of Email + Last four characters of First Name | MichJohn |
| First four characters of First Name + First four characters of Last Name | John |
| First four characters of Email + Last four characters of Last Name + | johnhael |

Custom schedule report settings

This section explain about custom schedule report settings to customize the **Schedule Template** and about **Predefiend Variable**.

Mail template customization

- Mail template define the text which is sent via email to the report server user.
- Initially the default template will displayed in the mail template and if the custom password is enable the user can add the password hint in the mail template if necessary
- And also you can customize the email template with your preference.

![Customize Email Template](/static/assets/on-premise/images/manage-schedule/manage-report-schedules/customize-email-template.png)

Mail template predefined variables

By selecting the variable the scheduled user variable is assigned directly from the server to the mail template

- {Full Name}: Full name of the recipient
- {First Name}: First name of the recipient
- {Last Name}: Last name of the recipient
- {Schedule Name}: Name of the schedule
- {Report Name}: Scheduled report name
- {Report Link}: Link to the scheduled report
- {Export Format}: Export file format which is chosen the schedule
- {Organization Name}: [Organization name](#) is retrieved from the site setting

Manage users

The users belongs to the **System Administrator** group can perform the following operations:

- Add, edit, and delete users
- Activate or deactivate users
- Manage permissions
- Assign users to groups

This section explains how to perform the above operation in the Bold Reports On-Premise. To manage the users, click the **User Management** option in Report Server left-side panel and navigate to **Users** tab as in following image.

![Open User Management settings](/static/assets/on-premise/images/manage-users-and-groups/users/user-management.png)

Add new users

The following steps are involved in adding new users to the Report Server:

1. In the **User Management** page, click **New User** and then **Create User** from menu list.
2. The **Add User** dialog will be shown as like in the following image.

![Add new user dialog page](/static/assets/on-premise/images/manage-users-and-groups/users/add-user.png)

3. Fill the form with Username, Email address, First name, Last name, and click **Add**.
4. New account will be created for the user.
5. An account activation email will be sent to the email address with activation link to activate the Report Server account.

The email setting must be enabled to send email notification. See [Email Settings](#) for configuration details.

The activation link sent to the user is valid only for 3 days. If the user have not activated within the 3 days, a new activation link can be send to the user from the user edit page. See [Activate Users](#) for more details.

The account will be in inactive status till the user activates the account using the activation email.

Edit users

You can edit user profile details such as First Name, Last Name, Email address, Phone Number, and Profile Picture.

![Edit User details](/static/assets/on-premise/images/manage-users-and-groups/users/edit-user.png)

The login password for the user can be edited as shown in the following image.

![Edit user password](/static/assets/on-premise/images/manage-users-and-groups/users/edit-password.png)

Delete users

Delete user from the Report Server when the user no longer requires the access. you can delete an user from the **User Management** page or from the profile edit page.

From user management page

Select the user from users list grid and click the **Delete User** option.

Manage users

Activate users

![Delete user from user management page](/static/assets/on-premise/images/manage-users-and-groups/users/delete-user-1.png)

From user edit page

Click an user from the users list grid and click **Delete User**.

![Delete user from user edit page](/static/assets/on-premise/images/manage-users-and-groups/users/delete-user-2.png)

Activate users

1. From the Users list, click the user name and it opens the user profile details page.
2. Click the **Edit Profile** option.
3. Activate the inactive users by clicking the **Activate User** button in the user edit page.

![Activate user account](/static/assets/on-premise/images/manage-users-and-groups/users/activate-user.png)

4. This will send an account activation email to the user with an activation link to activate the account. The activation link will be valid only for 3 days.
5. If the user has not received the activation email or missed to activate the account within 3 days, then the **System Administrator** should resend the activation email to the user as in the following image.

![Resend Activation code to activate user account](/static/assets/on-premise/images/manage-users-and-groups/users/resend-user-activation-link.png)

Deactivate users

Users can be deactivated at any time. Once deactivated, the user cannot log into the Report Server.

1. From the Users list page, click the user name, the user profile details page will open.
2. Click the **Edit Profile** option.
3. Choose inactive option from the status drop-down.

![Deactivate user account](/static/assets/on-premise/images/manage-users-and-groups/users/deactivate-user.png)

Assign users to group

Users can be assigned to one or many groups from the user management page.

1. Select and user from **User Management** and click the **Assign Group** option.

![Assign group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users.png)

2. Select the group name to assign the user to an existing group. Then click **Add**.

![Assign existing group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users-1.png)

3. To create a new group, click the **New Group** option. Specify the group name and click **Add**.

![Assign new group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users-2.png)

All the users in the group will have the permissions of assigned group.

Manage permissions

Check the [Manage Permissions](#) section to learn more about managing permissions to users.

Manage Users

This section explains on how to add, edit, activate, deactivate, delete users and also on how to manage the permissions and assign users to groups in the Bold Reports Server.

Users can only be added/edited/deleted by the users, belonging to the **System Administrator** group.

![Manage Users](/static/assets/on-premise/images/manage-users-and-groups/users/manage-users.png)

Add new users

New users can be added to the Bold Reports Server individually or in bulk using CSV import

Add individual users

To add new users to the Bold Reports Server, click on **New User** and then **Create User** from the User Management page.

The **Add User** dialog will be shown as like in the image below.

![Add User](/static/assets/on-premise/images/manage-users-and-groups/users/add-user.png)

Fill the form with Email address, First name and Last name and click on **Add**.

New account will be created for the user and an account activation email will be sent to the email address with activation link to activate the Bold Reports Server account.

The activation link sent to the user will be valid only for 2 days and if the user have not activated within the 2 days, a new activation link can be sent to the user from the user edit page. Check [Activate Users](#) section for more details.

The account will be in inactive status till the user activates the account from the activation email.

Edit users

User profile details can be edited from the users edit page as shown in the below image.

![Edit User](/static/assets/on-premise/images/manage-users-and-groups/users/edit-user.png)

First Name, Last Name, Phone number and the login password for the user can be edited by the user belonging to the 'System Administrator' group.

Delete users

Users can be deleted from the Bold Reports On-Premise when the user no longer requires the access.

Users can be deleted from the user management page or from the edit page.

From user management page

![Delete user from user management page](/static/assets/on-premise/images/manage-users-and-groups/users/delete-user-1.png)

From user edit page

![Delete user from user edit page](/static/assets/on-premise/images/manage-users-and-groups/users/delete-user-2.png)

Deactivate users

Users can be deactivated at any time. Once deactivated, the user cannot log into the Bold Reports On-Premise.

To deactivate a user, select inactive from the status dropdown in the user edit page.

![Deactivate user account](/static/assets/on-premise/images/manage-users-and-groups/users/deactivate-user.png)

Activate users

Inactive users can be activated by clicking on the **Resend Activation Code** button in the user edit page.

This will send an account activation email to the user with an activation link to activate the account and again this activation link will be valid only for 2 days.

If the user has not received the activation email within 2 days or missed to activate the account, the **System Administrator** has to resend the activation email to the user.

![Activate user account - Resend Activation code](/static/assets/on-premise/images/manage-users-and-groups/users/activate-user-resend.png)

Manage permissions

Check the [Manage Permissions](#) section to learn how to manage permissions to an user.

Assign users to group

Users can be assigned to one or many groups from the user management page.

![Assign group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users.png)

Users can be assigned to an existing group.

![Assign existing group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users-1.png)

A new group can also be created at this time and the selected users can be assigned to the new group.

![Assign new group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users-2.png)

All the users in the group will have the permissions of assigned group.

Manage Users

This section explains on how to add, edit, activate, deactivate, delete users and also on how to manage the permissions and assign users to groups in the Bold Reports Server.

Users can only be added/edited/deleted by the users, belonging to the **System Administrator** group.

![Manage Users](/static/assets/on-premise/images/manage-users-and-groups/users/manage-users.png)

Add new users

New users can be added to the Bold Reports Server individually or in bulk using CSV import

Add individual users

To add new users to the Bold Reports Server, click on **New User** and then **Create User** from the User Management page.

The **Add User** dialog will be shown as like in the image below.

![Add User](/static/assets/on-premise/images/manage-users-and-groups/users/add-user.png)

Fill the form with Email address, First name and Last name and click on **Add**.

New account will be created for the user and an account activation email will be sent to the email address with activation link to activate the Bold Reports Server account.

The activation link sent to the user will be valid only for 2 days and if the user have not activated within the 2 days, a new activation link can be sent to the user from the user edit page. Check [Activate Users](#) section for more details.

The account will be in inactive status till the user activates the account from the activation email.

Edit users

User profile details can be edited from the users edit page as shown in the below image.

![Edit User](/static/assets/on-premise/images/manage-users-and-groups/users/edit-user.png)

First Name, Last Name, Phone number and the login password for the user can be edited by the user belonging to the 'System Administrator' group.

Delete users

Users can be deleted from the Bold Reports On-Premise when the user no longer requires the access. Users can be deleted from the user management page or from the edit page.

From user management page

![Delete user from user management page](/static/assets/on-premise/images/manage-users-and-groups/users/delete-user-1.png)

From user edit page

![Delete user from user edit page](/static/assets/on-premise/images/manage-users-and-groups/users/delete-user-2.png)

Deactivate users

Users can be deactivated at any time. Once deactivated, the user cannot log into the Bold Reports On-Premise.

To deactivate a user, select inactive from the status dropdown in the user edit page.

![Deactivate user account](/static/assets/on-premise/images/manage-users-and-groups/users/deactivate-user.png)

Activate users

Inactive users can be activated by clicking on the **Resend Activation Code** button in the user edit page.

This will send an account activation email to the user with an activation link to activate the account and again this activation link will be valid only for 2 days.

If the user has not received the activation email within 2 days or missed to activate the account, the **System Administrator** has to resend the activation email to the user.

Import users from CSV

Manage permissions

![Activate user account - Resend Activation code](/static/assets/on-premise/images/manage-users-and-groups/users/activate-user-resend.png)

Manage permissions

Check the [Manage Permissions](#) section to learn how to manage permissions to an user.

Assign users to group

Users can be assigned to one or many groups from the user management page.

![Assign group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users.png)

Users can be assigned to an existing group.

![Assign existing group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users-1.png)

A new group can also be created at this time and the selected users can be assigned to the new group.

![Assign new group to selected users](/static/assets/on-premise/images/manage-users-and-groups/users/assign-group-to-users-2.png)

All the users in the group will have the permissions of assigned group.

Import users from CSV

You can add large number of users to Report Server by importing users from a CSV file. To automate the process of adding large number of users, download the CSV template file and add the users to it, then import the file.

1. In the Report Server page, navigate to the user management page by clicking the **User Management** icon.

![Open user management page](/static/assets/on-premise/images/manage-users-and-groups/users/user-management.png)

2. Click the **New User** option shown at right of the panel.
3. From the drop-down list, click **Import from CSV** option.

![Select Import from CSV from New User drop down](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-csv/import-csv-option.png)

4. This will open the Import Users page.

Download users CSV template file

To download the CSV user template, select the **Download Template** option from Import Users page as in the following image.

![Download CSV template option](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-csv/download-template.png)

CSV template requirements

The first row in the CSV template represents the column heading. Bold Reports On-Premise assumes that the data from the second line in the file represents the user. The following columns are considered as mandatory in the downloaded CSV file.

- Username
- Email address
- Full Name
- Password - If the account activation configured with Automatic, then password field should be filled. Otherwise you can leave the password field as empty.

Report Server has Automatic or Email based account activation.

Add or import users from CSV template

1. Download CSV template.
2. Add users in the CSV file and save the CSV file.

![Add users into CSV template file](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-csv/add-users-to-csv.png)

3. Click the browse button, select template file and click upload.

![Upload users CSV template file](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-csv/upload-csv.png)

4. After uploading the file, the user details will be shown in the grid as in the following image.

![Uploaded user detail shown in grid](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-csv/csv-imported-grid-view.png)

If the account activation configured with Email activation, the activation mail will be send to the user's mail Id. Otherwise the user will be automatically activated.

5. After uploaded the users in Report Server, the results are displayed as follows.

![Import success message](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-csv/csv-import-success.png)

6. Click the Import and Activate to import the users to Report Server.

Import users from CSV

You can add large number of users to Bold Report On-Premise by importing users from a CSV file. To automate the process of adding large number of users, download the CSV template file and add the users to it, then import the file.

You can navigate to user management page by click **User Management** drop down under the **Admin** menu as below.

![Import from CSV - Navigation](/static/assets/on-premise/images/manage-users-and-groups/users/usermanagement.png)

Add users from CSV file

In Bold Reports On-Premise, click **Import from csv**.

![Import from CSV - Goto Import CSV users page](/static/assets/on-premise/images/manage-users-and-groups/users/goto-import-csv-users.png)

CSV template requirements

The first row in the CSV template represents the column heading. Bold Reports On-Premise assumes that the data from the second line in the file represents the user.

We have the following columns are considered as mandatory in the downloaded CSV file.

- Email address
- Full Name

Follow the below steps to add users using the CSV template

1. Download CSV template.

![Import from CSV - Download CSV template](/static/assets/on-premise/images/manage-users-and-groups/users/csv-import.png)

2. Add users in the CSV file.

![Import from CSV - Add users into CSV file](/static/assets/on-premise/images/manage-users-and-groups/users/csv-import-add-users.png)

3. Save the CSV file and upload it.

![Import from CSV - Upload CSV file](/static/assets/on-premise/images/manage-users-and-groups/users/csv-import-upload.png)

4. Once the file is uploaded the user details will be shown in the grid as like in the below image.

![Import from CSV - User detail in grid](/static/assets/on-premise/images/manage-users-and-groups/users/csv-import-grid.png)

5. After uploaded the users in Bold Reports On-Premise the results are displayed as below.

![Import from CSV - Success Message](/static/assets/on-premise/images/manage-users-and-groups/users/import-csv-users-confirmation.png)

Import Active Directory users

Users belonging to the System Administrator group can import Active Directory users to the Report Server.

Active Directory connection must be configured in the [Active Directory Settings](#) for importing users.

This section explains how to search and import users from Active Directory into the Bold Reports On-Premise.

Search users

Initially, any Active Directory user cannot be displayed until searching for users.

1. Search the Active Directory users with anyone of the following properties:
 - User name
 - First name
 - Last name
 - Email address
 - Display name
2. The Report Server lists the search results in grid as shown in the following image.

![Search Active Directory users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-active-directory/search-user.png)

The search result will be based on "starts with" query.

3. Choose the users to import into the Report Server.

A maximum of 1000 users will be searched and pulled from Active Directory in a single request.

Import users

To import the Active Directory users into the Report Server, choose the users from the list and click **Import and Activate** at the top-right corner. After importing the Active Directory users, a confirmation message will be displayed as shown in the following image.

![Success message after imported the Active Directory users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-active-directory/user-imported.png)

Duplicate users

Users who has the same username or email address as that of the Report Server users(who are already present) will be marked as duplicate users. It will not be allowed to import into the Report Server.

![Duplicated Active Directory users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-active-directory/duplicated-users.png)

Modify Active Directory connection

To modify Active Directory configuration settings, click **Modify** as shown in the following image.

![Modify Active Directory configuration](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-active-directory/modify-configuration.png)

Azure Active Directory User Import

This section explains how to search and import users from Azure Active Directory into the Bold Reports On-Premise.

Azure Active Directory connection has to be configured in the [Azure Active Directory Settings](#) in the General page for importing users.

Users belonging to the **System Administrator** group only can import Azure Active Directory users into the Bold Reports.

Search Users

Initially, any Active Directory users cannot be displayed until searching for the user.

You can search the Azure Active Directory users with any one of the below properties and choose them to import into the Bold Reports.

- Email Address
- Full name

A maximum of 1000 users will be searched and pulled from Azure Active Directory in a single request.

Bold Reports will list the search results of the users in the grid as shown in the below figure.

The search result will be based on "starts with" query.

![Import Users from Azure Active Directory Server](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-azure-active-directory/Search-Azure-Active-Directory-User.png)

Import Users

To import the Azure Active Directory users into the Bold Reports, you have to choose the users from the list and click on the **Import and Activate** button at the top right corner.

Bold Reports will import the chosen users and a confirmation message will be displayed as shown in the below figure.

![Success message after imported the Azure Active Directory users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-azure-active-directory/Azure-Active-Directory-User-imported.png)

Duplicate Users

Azure Active Directory users who has the same email address as that of the Bold Reports users(who are already present) will be marked as duplicate users and will not be allowed to import into Bold Reports.

![Duplicated Azure Active Directory Users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-azure-active-directory/Duplicated-Azure-Active-Directory-Users.png)

Modify Azure Active Directory Connection

To modify Azure Active Directory configuration settings, click on the **Modify** link as below

![Modify Azure Active Directory Configuration](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-azure-active-directory/Modify-Azure-Active-Directory-Configuration.png)

User Import from a Database

This section explains how to import users from Database into the Bold Reports On-Premise.

Users belonging to the System Administrator group only can import users from database into the Bold Reports On-Premise.

Listing Database Users

To add new users to the Bold Reports On-Premise, click on **New User** and then **Import from Database** from the User Management page.

![Add New Users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-database/add-new-users.png)

The link will redirect to another page that will look like below.

![Import Users from Database - Home](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-database/import-users-home.png)

Select Users and Import

After selecting columns the data retrieved from database will be shown in Grid. Select the users to be imported and click on **Import Users** to import the users.

![Import Selected Users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-database/import-users-grid-selection-new.png)

Bold Reports will import the chosen users and a confirmation message will be displayed as shown in the below image.

![Success message after imported the Database users](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-database/Database-User-imported-new.png)

Modify Existing Database Connection

To modify Existing Database configuration settings, click on the **Modify** link as below

![Modify Database Configuration](/static/assets/on-premise/images/manage-users-and-groups/users/import-from-database/Modify-Database-settings-Configuration.png)

Synchronize Active Directory users

This section explains how to synchronize the imported Active Directory users details with the Active Directory.

Before synchronizing the Active Directory users, follow the given steps:

1. Configure the [Active Directory Settings](#).
2. [Import Active Directory Users](#) to the Bold Reports On-Premise.

Navigate to the user synchronization page from users page by clicking **Active Directory Synchronization** button as shown in the following image.

![Active Directory Synchronization button selection](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/active-directory-synchronize-option.png)

Synchronize users

Already imported Active Directory Users are displayed in a grid as shown in the following image.

![Active Directory imported users list grid](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/imported-user-list.png)

Choose the users you want to synchronize and click **Synchronize** at the top.

![Active Directory synchronize button](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/synchronize-button.png)

Synchronization procedure

- The user details such as username, first name, last name, email address, and contact number are synchronized with the Active Directory Server.
- Report Server will delete the user if the user has deleted from the Active Directory Server.
- After synchronization completes, the number of users modified, deleted, and duplicated will be shown in the success message box.

![Active Directory synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/synchronize-completed.png)

Duplicate users

The users who has the same username or email address as that of the Report Server users(who are already present) will be marked as duplicate users. It will not be allowed to synchronize into Active Directory.

![Displays duplicated users in synchronization](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/duplicate-users.png)

Synchronize Active Directory users

This section explains how to synchronize the imported Active Directory users details with the Active Directory.

Before synchronizing the Active Directory users, follow the given steps:

1. Configure the [Active Directory Settings](#).
2. [Import Active Directory Users](#) to the Bold Reports On-Premise.

Navigate to the user synchronization page from users page by clicking **Active Directory Synchronization** button as shown in the following image.

![Active Directory Synchronization button selection](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/active-directory-synchronize-option.png)

Synchronize users

Already imported Active Directory Users are displayed in a grid as shown in the following image.

![Active Directory imported users list grid](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/imported-user-list.png)

Choose the users you want to synchronize and click **Synchronize** at the top.

![Active Directory synchronize button](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/synchronize-button.png)

Synchronization procedure

- The user details such as username, first name, last name, email address, and contact number are synchronized with the Active Directory Server.
- Report Server will delete the user if the user has deleted from the Active Directory Server.
- After synchronization completes, the number of users modified, deleted, and duplicated will be shown in the success message box.

![Active Directory synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/synchronize-completed.png)

Duplicate users

The users who has the same username or email address as that of the Report Server users(who are already present) will be marked as duplicate users. It will not be allowed to synchronize into Active Directory.

![Displays duplicated users in synchronization](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-active-directory-users/duplicate-users.png)

Azure Active Directory User Synchronization

This section explains how to synchronize the imported Azure Active Directory users details with the Azure Active Directory.

Before synchronizing the Azure Active Directory users, follow the given steps:

1. Configure [Azure Active Directory Settings](#).
2. Import users from the Azure Active Directory to the Bold Reports by referring the following link [Active Directory User Import](#).

You can navigate to the user synchronization page from users page as shown in the below figure.

![Azure Active Directory Synchronization Link](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-azure-active-directory-users/azure-user-synchronisation-navigation-button.png)

Synchronize Users

Bold Reports will list the Azure Active Directory users that are already imported as shown in the below figure.

![Azure Active Directory Imported user list](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-azure-active-directory-users/imported-azure-user-list.png)

Only users imported from the Azure AD configured in this organization are listed here.

Choose the users you want to synchronize and click on **Synchronize** at the top.

![Synchronize button](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-azure-active-directory-users/Azure-Synchronize-button.png)

Synchronization procedure

- Bold Reports will synchronize the user details - username, first name, last name, email address, contact number with the Azure Active Directory Server.
- Bold Reports will delete the user if the user has deleted from the Azure Active Directory Server.

After synchronization completes, the number of users modified, deleted, duplicated will be shown in the success message box as shown in the below figure.

![Synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-azure-active-directory-users/Azure-Synchronization-Confirmation-window.png)

Duplicate Users

Azure Active Directory users who has the same email address as that of the Bold Reports users(who are already present) will be marked as duplicate users and will not be allowed to synchronize into Active Directory.

![Display Duplicated users](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-azure-active-directory-users/display-azure-duplicate-message.png)

Synchronization of Imported Users From the Existing Database

This section explains how to synchronize the imported existing database users details with the Existing database.

You can navigate to the user synchronization page from users page as shown in the below figure.

![Imported Database Users Synchronization Link](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-existing-database-users/user-synchronisation-navigation-button-for-importdb.png)

Synchronize Users

Bold Reports will list the Imported Database users that are already imported as shown in the below figure.

![Imported user list from Existing Database](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-existing-database-users/Imported-db-users-list.png)

Only users imported from the database configured in this organization are listed here.

Choose the users you want to synchronize and click on **Synchronize** at the top.

![Synchronize button](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-existing-database-users/synchronize-button-of-dbusers.png)

Synchronization procedure

- Bold Reports will synchronize the user details - first name, last name, email address, contact number with the Existing Database.
- Bold Reports will delete the user if the user has deleted from the Existing Database.

After synchronization completes, the number of users modified, deleted, duplicated will be shown in the success message box as shown in the below figure.

![Synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-existing-database-users/Synchronization-Confirmation-window-of-importdb.png)

Duplicate Users

Existing Database users who has the same email address as that of the Bold Reports users(who are already present) will be marked as duplicate users and will not be allowed to synchronize with the imported existing database users.

![Display Duplicated users](/static/assets/on-premise/images/manage-users-and-groups/users/synchronize-existing-database-users/display-duplicate-message-of-importdb.png)

Manage Groups

This section explains on how to add, edit, delete groups and also on how to assign users and manage permissions to groups in the Bold Reports On-Premise.

Groups is a collection of users to which permissions can be assigned.

![Manage Groups](/static/assets/on-premise/images/manage-users-and-groups/groups/manage-groups.png)

Add new group

To add new group to the Bold Reports On-Premise, click on **New Group** in the groups management page.

New groups can be added by providing name and description(optional) for the group.

![Add Groups](/static/assets/on-premise/images/manage-users-and-groups/groups/add-group.png)

Fill the form with name and description and click on **Add**. New group will be created and you can [add users](#) or [manage permissions](#) for it.

Edit group

Group Information can be edited from the group's edit page.

![Edit Group](/static/assets/on-premise/images/manage-users-and-groups/groups/edit-group.png)

Group name and description can be edited in the group edit page. In addition to that, users can also be assigned or removed from the group in this page.

Delete group

Groups can be deleted if it is no longer needed. You cannot delete the **System Administrator** group.

From group management page

![Delete group from group management page](/static/assets/on-premise/images/manage-users-and-groups/groups/delete-group-1.png)

From group edit page

![Delete group from group edit page](/static/assets/on-premise/images/manage-users-and-groups/groups/delete-group-2.png)

Assign users

Users can be assigned to the selected group there by assigning the permissions of the group to the users.

![Assign Users](/static/assets/on-premise/images/manage-users-and-groups/groups/assign-users-to-groups.png)

Users can also be removed from the group if the user no longer needs the permissions of the group. Click on Remove next to the user in the group edit page to remove the user from the group.

![Assign Users](/static/assets/on-premise/images/manage-users-and-groups/groups/remove-users-from-group.png)

Manage permissions

Check the [Manage Permissions](#) section to learn how to manage permissions to a group.

Manage Groups

This section explains on how to add, edit, delete groups and also on how to assign users and manage permissions to groups in the Bold Reports On-Premise.

Groups is a collection of users to which permissions can be assigned.

![Manage Groups](/static/assets/on-premise/images/manage-users-and-groups/groups/manage-groups.png)

Add new group

To add new group to the Bold Reports On-Premise, click on New Group in the groups management page.

New groups can be added by providing name and description(optional) for the group.

![Add Groups](/static/assets/on-premise/images/manage-users-and-groups/groups/add-group.png)

Fill the form with name and description and click on Add. New group will be created and you can [add users](#) or [manage permissions](#) for it.

Edit group

Group Information can be edited from the group's edit page.

![Edit Group](/static/assets/on-premise/images/manage-users-and-groups/groups/edit-group.png)

Group name and description can be edited in the group edit page. In addition to that, users can also be assigned or removed from the group in this page.

Delete group

Groups can be deleted if it is no longer needed. You cannot delete the System Administrator group.

From group management page

![Delete group from group management page](/static/assets/on-premise/images/manage-users-and-groups/groups/delete-group-1.png)

From group edit page

![Delete group from group edit page](/static/assets/on-premise/images/manage-users-and-groups/groups/delete-group-2.png)

Assign users

Users can be assigned to the selected group there by assigning the permissions of the group to the users.

![Assign Users](/static/assets/on-premise/images/manage-users-and-groups/groups/assign-users-to-groups.png)

Users can also be removed from the group if the user no longer needs the permissions of the group. Click on **Remove** next to the user in the group edit page to remove the user from the group.

![Assign Users](/static/assets/on-premise/images/manage-users-and-groups/groups/remove-users-from-group.png)

Manage permissions

Check the [Manage Permissions](#) section to learn how to manage permissions to a group.

Import Active Directory Groups

The users belonging to the System Administrator group can import Active Directory groups to the Report Server.

Active Directory connection must be configured in the [Active Directory Settings](#) for importing groups.

This section explains how to search and import groups from the Active Directory into the Bold Reports On-Premise.

Search groups

Initially, any Active Directory group cannot be displayed until searching for the group.

1. Search the Active Directory groups with any one of the following properties:
 - Group name
 - Group description

The Report Server lists the search results in grid as shown in the following image.

![Search Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-active-directory/searched-groups.png)

The search result will be based on "starts with" query.

2. Choose the groups to import into the Report Server.

A maximum of 1000 groups will be searched and pulled from Active Directory in a single request.

Import groups

To import the Active Directory groups into the Report Server, choose the groups from the list and click **Import Groups** at the top-right corner. After importing the Active Directory groups, a confirmation message will be displayed as shown in the following image.

![Success message after imported the Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-active-directory/group-import-success.png)

Duplicate groups

Groups that has the same name as that of the Report Server groups (that are already present) will be marked as duplicate groups. It will not be allowed to import into the Report Server.

![Duplicated Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-active-directory/duplicated-groups.png)

Import Active Directory Groups

The users belonging to the System Administrator group can import Active Directory groups to the Report Server.

Active Directory connection must be configured in the [Active Directory Settings](#) for importing groups.

This section explains how to search and import groups from the Active Directory into the Bold Reports On-Premise.

Search groups

Initially, any Active Directory group cannot be displayed until searching for the group.

1. Search the Active Directory groups with any one of the following properties:
 - Group name
 - Group description

The Report Server lists the search results in grid as shown in the following image.

![Search Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-active-directory/searched-groups.png)

The search result will be based on "starts with" query.

2. Choose the groups to import into the Report Server.

A maximum of 1000 groups will be searched and pulled from Active Directory in a single request.

Import groups

To import the Active Directory groups into the Report Server, choose the groups from the list and click **Import Groups** at the top-right corner. After importing the Active Directory groups, a confirmation message will be displayed as shown in the following image.

![Success message after imported the Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-active-directory/group-import-success.png)

Duplicate groups

Groups that has the same name as that of the Report Server groups (that are already present) will be marked as duplicate groups. It will not be allowed to import into the Report Server.

![Duplicated Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-active-directory/duplicated-groups.png)

Azure Active Directory Group Import

This section explain on how to search and import groups from Azure Active Directory into the Bold Reports On-Premise.

Azure Active Directory connection has to be configured in the [Azure Active Directory Settings](#) in the General page for importing groups.

Users belonging to the System Administrator group only can import Azure Active Directory groups into the Bold Reports On-Premise.

Search Groups

Initially, any Active Directory groups cannot be displayed until searching for the group.

You can search the Azure Active Directory groups with any one of the below properties and choose them to import into Bold Reports On-Premise.

- Group name

A maximum of 1000 groups will be searched and pulled from Azure Active Directory in a single request.

Bold Reports On-Premise will list the search results of the groups in the grid as shown in the below figure.

![Import groups from Azure Active Directory Server](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-azure-active-directory/Searched-azure-groups-list.png)

Import Groups

To import the Azure Active Directory groups into the Bold Reports On-Premise, you have to choose the groups from the list and click on the Import groups button at the top right corner.

Bold Reports On-Premise will import the chosen groups and a confirmation message will be displayed as shown in the below figure.

![Success message after imported the Azure Active Directory groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-azure-active-directory/Azure-Active-Directory-group-import-success-window.png)

The success message box explains the users who all are get imported/not imported into the Bold Reports On-Premise.

Duplicate Groups

Azure Active Directory groups who has the same group name as that of the Bold Reports On-Premise groups(which are already present) will be marked as duplicate groups and will not be allowed to import into Bold Reports On-Premise.

![Duplicated Azure Active Directory Groups](/static/assets/on-premise/images/manage-users-and-groups/groups/import-from-azure-active-directory/Azure-Active-Directory-Duplicate-group.png)

Synchronize Active Directory Group

This section explains how to synchronize the imported Active Directory groups and its users with the Active Directory.

Before synchronizing the Active Directory groups, follow the given steps:

- * Configure the [Active Directory Settings](#).
- * [Import Active Directory Groups](#) to the Bold Reports On-Premise.

Navigate to the user synchronization page from users page by clicking the **Active Directory Synchronization** button as shown in the following image.

![Active Directory synchronization button selection](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/active-directory-synchronization.png)

Synchronize groups

Already imported Active Directory groups are displayed in a grid as shown in the following image.

![Active Directory imported group list](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/active-direcory-group-list.png)

Choose the groups you want to synchronize and click **Synchronize** at the top.

![Active Directory group synchronize button](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/synchronize-button.png)

Synchronization procedure

- The group details such as name and description are synchronized with the Active Directory Server.
- Report Server will delete the group if the group has deleted from the Active Directory Server.
- Deletes the user from Report Server group, if the user has been deleted from Active Directory Server group. Adds the user into Report Server, if a new user is added into the Active Directory group. If the new user is not present in the Report Server, then a new user account will be created and added into the group.
- After synchronization completes, the number of groups modified, deleted, and duplicated will be shown in the success message box.

![Synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/group-synchronization-success.png)

Duplicate groups

Groups with same group name as that of the Report Server groups(which is already present) will be marked as duplicate group. It will not be allowed to synchronize into Active Directory.

![Displays duplicated groups in synchronization](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/highlight-duplicate-group.png)

Synchronize Active Directory Group

This section explains how to synchronize the imported Active Directory groups and its users with the Active Directory.

Before synchronizing the Active Directory groups, follow the given steps:

- * Configure the [Active Directory Settings](#).
- * [Import Active Directory Groups](#) to the Bold Reports On-Premise.

Navigate to the user synchronization page from users page by clicking the **Active Directory Synchronization** button as shown in the following image.

![Active Directory synchronization button selection](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/active-directory-synchronization.png)

Synchronize groups

Already imported Active Directory groups are displayed in a grid as shown in the following image.

![Active Directory imported group list](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/active-direcory-group-list.png)

Choose the groups you want to synchronize and click **Synchronize** at the top.

![Active Directory group synchronize button](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/synchronize-button.png)

Synchronization procedure

- The group details such as name and description are synchronized with the Active Directory Server.
- Report Server will delete the group if the group has deleted from the Active Directory Server.
- Deletes the user from Report Server group, if the user has been deleted from Active Directory Server group. Adds the user into Report Server, if a new user is added into the Active Directory group. If the new user is not present in the Report Server, then a new user account will be created and added into the group.
- After synchronization completes, the number of groups modified, deleted, and duplicated will be shown in the success message box.

![Synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/group-synchronization-success.png)

Duplicate groups

Groups with same group name as that of the Report Server groups(which is already present) will be marked as duplicate group. It will not be allowed to synchronize into Active Directory.

![Displays duplicated groups in synchronization](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-active-directory-groups/highlight-duplicate-group.png)

Azure Active Directory Group Synchronization

This section explains how to synchronize the imported Azure Active Directory group and its users with the Azure Active Directory.

Before synchronizing the Azure Active Directory groups, follow the given steps:

1. Configure [Azure Active Directory Settings](#)
2. Import groups from the Azure Active Directory to the Bold Reports On-Premise by referring the following link [Active Directory Group Import](#).

You can navigate to the group synchronization page from groups page as shown in the below figure.

![Azure Active Directory Synchronization Link](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-azure-active-directory-groups/Azure-Group-Synchronization-navigation.png)

Synchronize Groups

Bold Reports On-Premise will list the Azure Active Directory groups that are already imported as shown in the below figure.

![Azure Active Directory Imported group list](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-azure-active-directory-groups/Azure-Active-Direcory-Group-list.png)

Choose the groups you want to synchronize and click on **Synchronize** at the top.

![Azure Active Directory Group Synchronize button](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-azure-active-directory-groups/Azure-Group-Synchronize-button.png)

Synchronization procedure

- Bold Reports On-Premise will update the group's name and description from the Azure Active Directory Server.
- Bold Reports On-Premise will delete the groups if the group has been deleted from the Azure Active Directory Server.
- Bold Reports On-Premise will delete the user from Bold Reports On-Premise group, if the user has been deleted from Azure Active Directory Server group. Bold Reports On-Premise will add the user into Bold Reports On-Premise, if a new user is added into the Azure Active Directory group. If the new user is not present in the Bold Reports On-Premise, then a new user account will be created in the Bold Reports On-Premise and will be added into the group.

After synchronization completes, the number of groups modified, deleted, duplicated will be shown in the success message box as shown in the below figure.

![Synchronization confirmation window](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-azure-active-directory-groups/Synchronize-azure-group-success-window.png)

Duplicate Groups

Azure Active Directory groups who has the same group name as that of the Bold Reports On-Premise groups(which are already present) will be marked as duplicate groups and will not be allowed to synchronize with Azure Active Directory.

![Display Duplicated groups](/static/assets/on-premise/images/manage-users-and-groups/groups/synchronize-azure-active-directory-groups/highlight-duplicate-azure-group.png)

Account Activation

Account Activation should be configured to activate a user or group before importing from an existing database. To manage the activation mode, click **Settings** in the Report Server left-side panel and navigate to the User tab as shown in following image.

![Open Account activation settings](/static/assets/on-premise/images/manage-users-and-groups/account-activation.png)

There are two account activation modes available in the Report Server.

Automatic activation

When users select **Automatic Activation** mode, user accounts will be activated automatically while adding users to the Report Server. But, account activation mails will not be sent.

Email activation

When users select **Email Activation** mode, user accounts will not be activated while adding users to the Report Server and an account activation email will be sent to the email address with activation link to activate the user management account.

The activation link sent to the user will be valid for only 3 days. If the user did not activate within 3 days, a new activation link will be sent again to the user from the user edit page. Refer to the activate users section for more details.

The account will be in inactive status until the user activates the account from the activation email.

Manage Permissions

Permissions can be managed only by the users belonging to the **System administrator** group.

Permissions can be directly added to both users and groups and can be classified in the **Access Mode – Entity – Scope** structure.

This section explains the access modes, entities, scopes, and how to manage the permissions for users and groups.

Access modes

- Read: Provides read permission for the chosen entity.
- Read and Write: Provides read and write permission for the chosen entity.
- Read, Write, and Delete: Provides read, write, and delete permission for the chosen entity.
- Read and Download: Provides read and download permission for the chosen entity.
- Read, Write, and Download: Provides read, write, and download permission for the chosen entity.
- Read, Write, Delete, and Download: Provides read, write, delete, and download permission for the chosen entity.
- Create: Provides permission to create the chosen entity.

Entity

- All Reports: Provides permission to access all the reports with the chosen access mode.
- Reports in Category: Provides permission to access reports in a specific category with chosen access mode.
- Specific Report: Provides permission to access a specific report with the chosen access mode.
- All Data Sources: Provides permission to access all data sources with the chosen access mode.
- Specific Data Source: Provides permission to access a specific data source with the chosen access mode.
- Specific File: Provides permission to access a specific file with the chosen access mode.
- All Categories: Provides permission to access all categories with the chosen access mode.
- Specific Category: Provides permission to access a specific category with the chosen access mode.
- All Schedules: Provides permission to access all schedules with the chosen access mode.
- Specific Schedule: Provides permission to access a specific schedule with the chosen access mode.

Scope

Choose scopes for the following entities only, other entities do not require scopes.

- Reports in Category: A specific category has to be chosen to provide access to the reports in that category.
- Specific Report: A specific report has to be chosen to provide access to it.
- Specific Data Source: A specific data source has to be chosen to provide access to it.
- Specific Category: A specific category has to be chosen to provide access to it.
- Specific Schedule: A specific schedule has to be chosen to provide access to it.

Create access only have the following scopes:

- * All Reports
- * Reports in Category
- * All Data Sources
- * All Datasets
- * All Schedules and All Categories

Manage users permissions

Manage Permissions page for the users can be accessed from any one of the following places.

1. Click on the Manage Permission icon for the respective users in the users grid on the user management page.

![Manage Permissions context menu for users](/static/assets/on-premise/images/manage-permissions/manage-permission-icon-user.png)

2. On the top right corner of the user profile edit page.

![Manage Permissions option on edit user page](/static/assets/on-premise/images/manage-permissions/edit-user.png)

3. Here, you will find both the permissions assigned directly to the user and the permissions that the user got inherited from the groups assigned with.

![List user permissions details](/static/assets/on-premise/images/manage-permissions/list-user-permissions.png)

4. Click **Add** in the Add Permission dialog box to add permissions to users as shown in the following image.

![Add permission to user](/static/assets/on-premise/images/manage-permissions/add-permission-to-user.png)

Note: By clicking on the check box **Add another**, you can add multiple permission to the groups.

Steps to add permission to users

1. Select the access mode.
2. Select the entity.
3. Select the scope if the access mode is not Create or if the entity is specific item type.
4. Click **Add** to add the framed permission to users.

Manage group permissions

Manage Permissions page for the group can be accessed from any one of the following places.

1. Click on the **Manage Permission** icon for the respective group in the groups grid on the group management page.

![Manage Permissions context menu for group](/static/assets/on-premise/images/manage-permissions/manage-permission-icon-group.png)

2. On the top right corner of the group edit page.

![Manage Permissions option on edit group page](/static/assets/on-premise/images/manage-permissions/edit-group.png)

3. Here, you will find the permissions assigned directly to the group. Refer to the following screenshot for **Manage Permissions** in the user page.

![List user permissions details](/static/assets/on-premise/images/manage-permissions/list-group-permissions.png)

4. Click **Add** in the Add Permission dialog box to add permissions to the group as shown in the following image.

![Add permission to group](/static/assets/on-premise/images/manage-permissions/add-permission-to-group.png)

Note: By clicking on the check box **Add another**, you can add multiple permission to the groups.

Steps to add permission to the group

1. Select the access mode.
2. Select the entity.
3. Select the scope if the access mode is not Create or if the entity is specific item type.
4. Click **Add** to add framed permission to the group.

Manage Categories

This section explains how to create, open, update, delete, and share permissions in categories on the Bold Reports On-Premise. Categories are accessible by users depends upon users's permission.

Category are displayed on the left side of report listing page.

![Category setting options](/static/assets/on-premise/images/manage-categories/category-setting-options.png)

Open category

To open the already created category, click any category in the left panel to view the reports grouped with it.

![Open already created categories](/static/assets/on-premise/images/manage-categories/open-category.png)

Create category

To add a new category, you should have **Create All Categories** permission.

Category can be added by the below ways,

1. While publishing the designed report to the Report Server, **New Category** option will be shown.

![Publish report](/static/assets/on-premise/images/manage-categories/publish-report.png)

2. While copying the reports from context menu of the report listing page.

![Publish report](/static/assets/on-premise/images/manage-categories/create-category.png)

Provide a name and description (optional) for the category and click **Add**.

![Add a new category for reports](/static/assets/on-premise/images/manage-categories/add-category.png)

Read, Write, Delete permission for that **Specific Category** is provided to the user who created the category.

Update category

To update the category name and description of the already created category, follow these steps:

1. Click the settings option available in the category.
2. Select **Update** from the context menu.
3. Modify the name or description and click **Update**.

![Update category name and description](/static/assets/on-premise/images/manage-categories/update-category.png)

Share category

You can set permissions to the categories to share with other users in the Report Server.

Steps to share a category

1. Click **Actions** in the category list context menu and select **Sharing Permissions**.

![Select Sharing Permissions option](/static/assets/on-premise/images/manage-categories/category-setting-options.png)

2. In the Sharing Permissions dialog, click **Share To** tab. By default, it will be open with **Share To** tab.
3. Select the permission access from the Select Access drop-down and select the users or groups to share the category.

![select the users or groups to share the category](/static/assets/on-premise/images/manage-categories/share-category.png)

4. After selecting the access, users, and groups, click **Share** button.

![Add category permission to share](/static/assets/on-premise/images/manage-categories/add-permission.png)

Only the user who created the category can share it with other users.

[Remove permission](#)

The user who created the category can remove the shared category permissions by following these steps:

1. In the Sharing Permissions dialog, click **Share With** tab. select the user or group to remove the permission.
2. Click **Remove** in the Actions column of the each permissions.

![Remove user or group shared permission](/static/assets/on-premise/images/manage-categories/remove-permission.png)

[Delete category](#)

You can delete a category from the Report Server by following these steps:

1. Click **Actions** in the category list context menu.
2. Select **Delete** to delete the category.

![Delete created category](/static/assets/on-premise/images/manage-categories/delete-category.png)

Category cannot be deleted when it has reports grouped in it.

[REST API Reference](#)

The following table illustrates the list of available APIs related to category in Bold Reports On-Premise.

| Action | HTTP Method | Endpoint | Description |
|----------------|-------------|-----------------------------------------|---------------------------------------------------------------------------------------|
| AddCategory | POST | /api/site/{tenant-name}/v1.0/categories | Adds a new category to the server. Category details must be passed as input. |
| UpdateCategory | PUT | /api/site/{tenant-name}/v1.0/categories | Updates the category in the server. Updated category details must be passed as input. |

| | | | |
|-----------------------------|--------|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>GetCategory</u> | GET | /api/site/{tenant-name}/v1.0/items | Returns the list of categories for current user. ItemType should be Category. |
| <u>IsCategoryNameExists</u> | POST | /api/site/{tenant-name}/v1.0/items/is-name-exists | Returns an item existence whether the given category name already exists or not in server. Category name and ItemType as Category should be passed in request body. |
| <u>GetCategoryDetail</u> | GET | /api/site/{tenant-name}/v1.0/items/{id} | Returns the specified category details from the server. Category item ID should be passed in path parameter. |
| <u>DeleteCategory</u> | DELETE | /api/site/{tenant-name}/v1.0/items/{id} | Deletes the specified category from the server. Category item ID should be passed in path parameter. |

Collaboration

The users can write comments on a report to share with other users who has access to the report. It is useful for tracking events and providing insights into those events. Also, users can add links to other reports or any other external websites.

When a comment is added to a report, users who have enabled notifications in their profile will be notified through email and system notifications. For more details, refer to [Notifications](#).

This section explains how to collaborate with other users in the Bold Reports On-Premise by commenting on reports.

Post a new comment

To post a new comment, open the report and click the comment icon in the top-right corner as shown in the following image.

![Comment icon](/static/assets/on-premise/images/collaboration/comment-button.png)

Type the comment in the text area and click **Post** to save the comment for the report.

![Click on the post option](/static/assets/on-premise/images/collaboration/comment-post.png)

![Comment saved](/static/assets/on-premise/images/collaboration/comment-saved.png)

Clipboard images can also be added along with the comments by simply copying an image and pasting in the text area.

Reply to a comment

To reply to a comment, click the **Reply** icon in the comment as shown in the following image.

![Comment reply icon](/static/assets/on-premise/images/collaboration/comment-reply-icon-click.png)

Type the reply in the text area and click **Reply** to save the reply for the comment in the Report.

![Click on reply option](/static/assets/on-premise/images/collaboration/comment-reply-button.png)

![Reply comment saved](/static/assets/on-premise/images/collaboration/comment-reply-saved.png)

You can also reply to the comment. This can be repeated for number of times.

Edit a comment

To edit a comment, click the **Actions** button to get more options about a comment or a reply and click **Edit** as shown in the following image.

![Edit comment menu option](/static/assets/on-premise/images/collaboration/comment-edit-button.png)

Edit the comment and click **Save** to save it.

![Edited comment save option](/static/assets/on-premise/images/collaboration/comment-edited.png)

![Edited comment saved](/static/assets/on-premise/images/collaboration/comment-edit-saved.png)

Delete a comment

To delete a comment, click the **Actions** button to get more options about a comment or a reply and click **Delete** as shown in the following image.

![Delete comment menu option](/static/assets/on-premise/images/collaboration/comment-delete-button.png)

Show parent comment of a reply

To know the parent comment of a reply or to know for which comment the reply has been posted, click the **Actions** button and click **Show Parent** as shown in the following image.

![Show parent comment icon](/static/assets/on-premise/images/collaboration/comment-show-parent-button.png)

On clicking, the parent comment is highlighted for the reply as follows.

![Click on show parent icon](/static/assets/on-premise/images/collaboration/comment-show-parent.png)

Mention users in the comment

Users can be mentioned in the comments to notify them about the comment through email.

Type @ followed by the user's name and from the list of possible names, select the user to mention them in the comment.

![Specify user name to mention in comment](/static/assets/on-premise/images/collaboration/mention-user.png)

![User name to mentioned in comment](/static/assets/on-premise/images/collaboration/user-mentioned.png)

The options such as post a new comment, reply to a comment, edit a comment, delete a comment, and show parent comment of a reply are applied to reports.

Notifications

You can configure the notifications settings to notify the users when any comment is added to the reports in the Bold Reports On-Premise. Notifications can be configured by both the System Administrator and user.

To change the notification settings, click the Setting option in the Report Server left-side panel and navigate to the Notification tab as shown in the following image.

![Notification settings page](/static/assets/on-premise/images/notifications/notifications-settings.png)

Admin notification settings

Configure how the users receive notifications for the report comments from the admin notification settings page.

![Admin notification settings](/static/assets/on-premise/images/notifications/admin-notifications-settings.png)

System notifications

To view the system notifications, click on the **Notification** icon in the Report Server left-side panel as shown in the following image.

![System notifications](/static/assets/on-premise/images/notifications/system-notifications.png)

Mail notifications

Enabling the mail notifications option will notify the users through email for comments.

Auto watch comments of created items

Enabling this will send notifications for comments on all the items created by users.

Auto watch comments of accessible items

Enabling this will send notifications for comments on all the items that are accessed by users.

Report schedule notification

Enabling and disabling report schedule button we can restrict the mail receiving for scheduled report

User schedule notification

Enabling and disabling user schedule button we can restrict the mail receiving for admin users

The following are the default and allowable notification settings configuration.

Default settings

This is the default settings applied to users when users are added to the Report Server. Users can change switch from this setting and make their own or inherit this setting at anytime in their profile edit page.

Allow/restrict settings

This is the master settings for the Report Server. Enabling or disabling any setting, will enable or disable it in the Report Server. This will override the default and user settings.

User notification settings

Configure how the current user receive notifications for comments from the user notification settings page. Users can navigate to this page from the profile edit page as shown in the following image.

![User notification settings](/static/assets/on-premise/images/notifications/user-notifications-settings-navigation.png)

Refer to the following image for changing the notification settings for the current user.

![User notification settings options](/static/assets/on-premise/images/notifications/user-notifications-settings.png)

Settings can be enabled, disabled, or inherited from the global settings that is the default settings of the Report Server.

Specific watch

Apart from auto watch of created and accessible item settings, users can also watch an item specifically.

![Report Specific watch](/static/assets/on-premise/images/notifications/report-specific-notifications.png)

Users can toggle between watch and unwatch for a report comment at anytime.

Authentication settings

You can configure Single Sign-On(SSO) in Bold Reports by using Authentication Settings. You can refer more details from the following sections.

Single Sign-On

Single Sign-On

Single Sign-On (SSO) is an authentication scheme that allows you to login into the multiple applications using single id and password. Single Sign-On(SSO) for Bold Reports can be configured with external authentication providers like WS-Federation (ADFS), OAuth 2.0 and Open ID identity specifications.

You can refer more details from the following sections to configure Single Sign-On for Bold Reports:

OAuth 2.0 - OneLogin in Bold Reports

Single Sign-On(SSO) with OAuth 2.0 authentication in Bold Reports

The Bold Reports application can be configured with OAuth 2.0 for Single Sign-On(SSO), so that the users can log in directly to Bold Reports application after authenticating using the OAuth 2.0.

Prerequisites

1. An account with an OAuth 2.0 provider.
2. Register the Bold Reports application in the OAuth 2.0 provider.

Steps to configure OAuth 2.0 in Bold Reports

1. To configure the OAuth 2.0 connection details, click the settings option in the Report Server left-side panel.
2. Navigate to the Authentication tab and OAuth 2.0 as shown in the following image

![Authentication](/static/assets/on-premise/images/authentication/single-sign-on/oauth/authentication-settings.png)

3. Provide the following details in the OAuth 2.0 settings of Bold Reports application.

| | |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authorization Endpoint | It is the endpoint in the provider to authorize the user. |
| Token Endpoint Method | It represents the request type to access the token endpoint. |
| Token Endpoint | It is the endpoint in the provider that generates the token. |
| User Information Endpoint Method | It is the endpoint in the provider used to get the user details. |
| User Information Endpoint | It represents the request type to access the user information endpoint. |
| Client ID | It is an unique identifier provided to each of the applications while registering in the providers. |
| Client Secret | It is a secret key that is used to authorize the applications. |
| Scopes | It is comma separated lists of identifiers that specifies the access privileges that are being requested from the provider. |
| Email | This must be the email of an admin account of the providers. |

![Oauth Authentication](/static/assets/on-premise/images/authentication/single-sign-on/oauth/oauth-authentication.png)

The previous mentioned similar steps are applicable to configure the OAuth 2.0 in User Management Server by logging into the URL {Bold Reports URL}/ums/administration/authentication with admin credential.

Following are the list of few OAuth 2.0 providers and that explains how to connect with the Bold Reports application.

- [Amazon Cognito](#)
- [Auth0](#)
- [Okta](#)
- [OneLogin](#)

The OAuth 2.0 is provided in both the Bold Reports sites and User Management Server. OAuth 2.0 can be handled for each sites individually in the settings page by disabling the option as in the following screenshot.

![Oauth Group Setting](/static/assets/on-premise/images/authentication/single-sign-on/oauth/enableoauth.png)

Single Sign-On(SSO) with OneLogin authentication in Bold Reports

Users can be added to the Bold Reports application using the OneLogin provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in OneLogin

This section explains how to perform Single Sign-On(SSO) for users in OneLogin with Bold Reports application.

This configuration has to be done in OneLogin website.

Prerequisites

- An admin account in OneLogin.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the OneLogin website with the admin account.
2. Click on Applications in the header menu.

![Application page](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/oneloginadmin.png)

3. Click on Add App button.

![Add Application page](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/oneloginaddapp.png)

4. Type the word OpenId Connector oidc in the search box and click on the result.

![Openidconnect](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/openidconnect.png)

5. Type the application name and click on Add App.
6. Save the application name in the Display Name.

![Applicationname](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/oneloginname.png)

7. Click the Configuration tab and save the Redirect URI's and Login Url. For Bold Reports mobile application, use the Mobile App Redirect URI and use the Redirect URI for the web application.

![Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/onelogin-save-redirect-uri.png)

8. The Redirect URI and Login URL is found under the OpenID Connect settings of your Bold Reports application as in the following screenshot.

![Redirecturi in setting](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/login-redirect-uri.png)

9. Click SSO tab, you will find the Client ID and Client Secret and use it in the Bold Reports application.

![OneLoginClient details](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/onelogin-client-secret.png)

After successful registration in OneLogin, save these settings in Bold Reports settings page to enable this authentication.

[Enable OneLogin authentication in Bold Reports](#)

Configure the settings in Bold Reports as in the following snap to enable the authentication using OneLogin.

![OneLogin settings](/static/assets/on-premise/images/authentication/single-sign-on/oauth/onelogin/configure-boldreport-onelogin.png)

The fields to be saved in the Bold Reports to enable the OneLogin for authentication is explained as below.

| | |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authorization Endpoint | It should be in the format https://subdomain.onelogin.com/oidc/auth Please refer here for more details. |
| Token Endpoint Method | POST |
| Token Endpoint | It should be in the format https://subdomain.onelogin.com/oidc/token Please refer here for more details. |
| User Information Endpoint Method | GET |
| User Information Endpoint | It should be in the format https://subdomain.onelogin.com/oidc/me Please refer here for more details. |
| Client ID | The Client ID you get after registered the Bold Reports application in OneLogin website. |
| Client Secret | The Client Secret you get after registered the Bold Reports application in OneLogin website. |
| Scopes | openid, profile, email |
| Email | This must be the email of an admin account of OneLogin website. |

Single Sign-On(SSO) with Okta authentication in Bold Reports

Users can be added to the Bold Reports application using the Okta provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in Okta

This section explains how to perform Single Sign-On for users in Okta with Bold Reports application.

This configuration has to be done in Okta website.

Prerequisites

- An admin account in Okta.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the Okta website with an **admin** account.

![Login Okta](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta-login.png)

2. Click **Applications** in the header menu.

![Click Application](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta-application.png)

3. Click **Add Application** button.

![Add Application](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta-add-application.png)

4. Click **web** and proceed with **Next**.

![Select Web Platform](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta-platform-web.png)

5. Fill in the following application details in the next page and click **Done**.

- Name
- Login Redirect URIs
- Logout Redirect URIs

![Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta-redirect-uri.png)

6. The Login redirect URIs must be the URI in the settings of your Bold Reports application as in the following snap.

Single Sign-On(SSO) with Amazon Cognito authentication in Bold Reports Enable Okta authentication in Bold Reports

![Login Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta/login-redirect-uri.png)

7. In the next page, you will get the **Client ID** and **Client Secret** at the bottom of the page along with the other details you filled in the previous page.

![Okta Client Credential](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta/okta-client-credential.png)

After successful registration in Okta, save these settings in Bold Reports settings page to enable this authentication.

Enable Okta authentication in Bold Reports

Configure the settings in Bold Reports as in the following snap to enable the authentication using Okta.

![Configure Bold Reports Okta](/static/assets/on-premise/images/authentication/single-sign-on/oauth/okta/configure-boldreport-okta.png)

The fields to be saved in the Bold Reports to enable the Okta for authentication is explained as follows.

| | |
|----------------------------------|----------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authorization Endpoint | It should be in the format \${baseUrl}/v1/authorize Please refer here for more details.. |
| Token Endpoint Method | POST |
| Token Endpoint | It should be in the format \${baseUrl}/v1/token Please refer here for more details. |
| User Information Endpoint Method | POST |
| User Information Endpoint | It should be in the format \${baseUrl}/v1/token Please refer here for more details. |
| Client ID | The Client ID you get after registered the Bold Reports application in Okta website. |
| Client Secret | The Client Secret you get after registered the Bold Reports application in Okta website. |
| Scopes | openid, profile, email |
| Email | This must be the email of an admin account of Okta website. |

Single Sign-On(SSO) with Amazon Cognito authentication in Bold Reports

Users can be added to the Bold Reports application using the Amazon Cognito provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in Amazon Cognito

This section explains how to perform Single Sign-On for users in Amazon Cognito with the Bold Reports application.

This configuration has to be done in Amazon Cognito website.

Prerequisites

- An admin account in Amazon Cognito.
- An user pool in Amazon Cognito.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the Amazon Cognito website with an admin account and open the console and then click **Manage User pool**.

![Click Manage User Pool](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/click-manage-user-pool.png)

2. Click **App Clients** under General Settings in the left side menu, and then add the **application**.

![Click App Client](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/cognito-click-app-client.png)

3. Save the App client name and click on **Create app client**.

![Create App Client](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/cognito-create-app-client.png)

4. Click on **Show details** to know the Client Secret.

![Cognito Client Secret](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/cognito-client-secret.png)

5. Make use of the **App client id** and **App client secret** as in the following screenshot.

![Cognito Credential](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/cognito-clientsecret-clientid.png)

6. Click on **App client settings** under App integration in the left side menu and add the **Callback URL(s)**.

![Cognito Call Back URL](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/congnito-call-back-url.png)

7. The Callback URL(s) must be the URI in the settings of your BoldReports application as in the following screenshot.

![Login Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/login-redirect-uri.png)

After successful registration in Amazon Cognito, save these settings in BoldReports settings page to enable this authentication.

Enable Amazon Cognito authentication in Bold Reports

Configure the settings in Bold Reports as in the following snap to enable the authentication using Amazon Cognito.

![Configure BoldReports](/static/assets/on-premise/images/authentication/single-sign-on/oauth/amazon-cognito/configure-boldreport-amazon-cognito.png)

The fields to be saved in the Bold Reports to enable the Amazon Cognito for authentication is explained as below.

| | |
|----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authorization Endpoint | It should be in the format https://AUTH_DOMAIN/oauth2/authorize Please refer here for more details. |
| Token Endpoint Method | POST |
| Token Endpoint | It should be in the format https://AUTH_DOMAIN/oauth2/token Please refer here for more details. |
| User Information Endpoint Method | GET |
| User Information Endpoint | It should be in the format https://AUTH_DOMAIN/oauth2/userinfo Please refer here for more details. |
| Client ID | The Client ID you get after registered the Bold Reports application in Amazon Cognito website. |
| Client Secret | The Client ID you get after registered the Bold Reports application in Amazon Cognito website. |
| Scopes | openid, profile, email |
| Email | This must be the email of an admin account of Amazon Cognito website. |

Single Sign-On(SSO) with Auth0 authentication in Bold Reports

Users can be added to the Bold Reports application using the Auth0 provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in Auth0

This section explains how to perform Single Sign-On for users in Auth0 with the Bold Reports application.

This configuration has to be done in Auth0 website.

Prerequisites

- An admin account in Auth0.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the Auth0 website with an **admin** account.

![Auth0 Login Admin Account](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/auth0-login-admin-account.png)

2. Click **Applications** in the left menu and then click **CREATE APPLICATION** button.

![Auth0 create Application](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/auth0-create-application.png)

3. Click **Regular Web Applications** and proceed with **Next**.

![Select Regular Web Application](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/select-regular-web-application.png)

4. The application will be registered and directed to the application details page. Use the **Client ID** and **Client Secret** in the BoldReports application.

![auth0 ClientId and Client Secret](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/auth0-clientid-client-secret.png)

5. Scroll down and save the **Allowed Callback URLs**.

![Auth0 Save Callback URL](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/auth0-save-callback-url.png)

6. This URL must be the redirect URI in the settings of your Bold Reports application as in the following screenshot.

![Lodin Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/login-redirect-uri.png)

7. Fill the details and save the changes.

After successful registration in Auth0, save these settings in Bold Reports settings page to enable this authentication.

[Enable Auth0 authentication in Bold Reports](#)

Configure the settings in Bold Reports as in the following snap to enable the authentication using Auth0.

![Configure Bold Report Auth0](/static/assets/on-premise/images/authentication/single-sign-on/oauth/auth0/configure-boldreport-auth0.png)

The fields to be saved in the Bold Reports to enable the Auth0 for authentication is explained as follows.

| | |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authorization Endpoint | It should be in the format https://YOUR_DOMAIN/authorize Please refer here for more details. |
| Token Endpoint Method | POST |
| Token Endpoint | It should be in the format https://YOUR_DOMAIN/oauth/token Please refer here for more details. |
| User Information Endpoint Method | GET |
| User Information Endpoint | It should be in the format https://YOUR_DOMAIN/userinfo Please refer here for more details. |
| Client ID | The Client ID you get after registered the Bold Reports application in Auth0 website. |
| Client Secret | The Client Secret you get after registered the Bold Reports application in Auth0 website. |
| Scopes | openid, profile, email |
| Email | This must be the email of an admin account of Auth0 website. |

[Single Sign-On\(SSO\) with OpenID Connect authentication](#)

The Bold Reports application can be configured with OpenID Connect for Single Sign-On (SSO), so that the users can log in directly to Bold Reports application after authenticating using the OpenID Connect.

[Prerequisites](#)

1. An account with an OpenID Connect provider.
2. Register the Bold Reports application in the OpenID Connect provider.

Steps to configure OpenID Connect in Bold Reports

1. To configure the OpenID Connect connection details, click the settings option in the Report Server left-side panel.
2. Navigate to the Authentication tab and OpenID Connect as shown in the following image
3. Provide the following details in the OpenID Connect settings of Bold Reports application.

| | |
|---------------|-----------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authority | It is the instance created for the user in the provider. |
| Client ID | It is an unique identifier provided to each of the applications while registering in the providers. |
| Client Secret | It is a secret key that is used to authorize the applications. |
| Identifier | It is the property name that holds the email address of the user in the serialized ID token. |

![OpenID Connect Authentication](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/openid-connect-authentication.png)

The previous mentioned similar steps are applicable to configure the OpenID Connect in User Management Server by logging into the URL {Bold Reports URL}/ums/administration/authentication with admin credential.

Following are the list of few OpenID Connect providers and that explains how to connect with the Bold Reports application.

- [Auth0](#)
- [Okta](#)
- [OneLogin](#)

The OpenID Connect is provided in both the Bold Reports sites and User Management Server. OpenID Connect can be handled for each sites individually in the settings page by disabling the option as in the following screenshot.

![OpenId Global Authentication Control](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/openid-global-authentication-control.png)

Single Sign-On(SSO) with OneLogin authentication

Users can be added to the Bold Reports application using the OneLogin provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in OneLogin

This section explains how to perform Single Sign-On for users in OneLogin with Bold Reports application.

This configuration has to be done in OneLogin website.

Prerequisites

- An admin account in OneLogin.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the OneLogin website with the **admin** account.
2. Click on **Applications** in the header menu.

![Click OneLogin Application](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/click-onelogin-application.png)

3. Click on **Add App** button.

![Add App in OneLogin](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/add-app-onelogin.png)

4. Type the word **OpenID Connect** or **oidc** in the search box and click on the result.

![Search OpenID Connect](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/search-openid-connect.png)

5. Type the application name and click on **Add App**
6. Save the application name in the **Display Name**

![Save Application](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/onelogin-save-application.png)

7. Click the Configuration tab and save the **Redirect URI's** and **Login Url**.

![Save Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/onelogin-save-redirect-uri.png)

8. The Redirect URI and Login URL is found under the OpenID Connect settings of your Bold Reports application as in the following screenshot.

Single Sign-On(SSO) with Okta authentication in Bold Reports Enable OneLogin authentication in Bold Reports

![Login Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/login-redirect-uri.png)

9. Click SSO tab, you will find the Client ID and Client Secret and use it in the Bold Reports application.

![Client Secret](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/onelogin-client-secret.png)

After successful registration in OneLogin, save these settings in Bold Reports settings page to enable this authentication.

Enable OneLogin authentication in Bold Reports

Configure the settings in Bold Reports as in the following snap to enable the authentication using OneLogin.

![Configure Bold Reports Openid OneLogin](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/onelogin/configure-boldreport-openid-onelogin.png)

The fields to be saved in the Bold Reports to enable the OneLogin for authentication is explained as below.

| | |
|---------------|----------------------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authority | It must be in the form https://subdomain.onelogin.com/oidc. Sub domain represents the OneLogin instance. |
| Client ID | The Client ID you get after registered the Bold Reports application in OneLogin website. |
| Client Secret | The Client Secret you get after registered the Bold Reports application in OneLogin website. |
| Identifier | preferred_username |

Single Sign-On(SSO) with Okta authentication in Bold Reports

Users can be added to the Bold Reports application using the Okta provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in Okta

This section explains how to perform Single Sign-On for users in Okta with the Bold Reports application.

This configuration has to be done in Okta website.

Prerequisites

- An admin account in Okta.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the Okta website with an **admin** account.

![Login Okta](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/okta-login.png)

2. Click **Applications** in the header menu.

![Click Application](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/okta-application.png)

3. Click **Add Application** button.

![Add Application](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/okta-add-application.png)

4. Click **web** and proceed with **Next**.

![Select Web Platform](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/okta-platform-web.png)

5. Fill in the following application details in the next page and click **Done**.

- Name
- Login Redirect URIs
- Logout Redirect URIs

![Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/okta-redirect-uri.png)

6. The Login redirect URIs must be the URI in the settings of your Bold Reports application as in the following snap.

![Login Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/login-redirect-uri.png)

7. In the next page, you will get the **Client ID** and **Client Secret** at the bottom of the page along with the other details you filled in the previous page.

![Okta Client Credential](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/okta-client-credential.png)

Enable Okta authentication in Bold Reports

Configure the settings in Bold Reports as in the following screenshot to enable the authentication using Okta.

![Configure Bold Report OpenID Okta](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/okta/configure-boldreport-openid-Okta.png)

The fields to be saved in the Bold Reports to enable the Okta for authentication is explained as follows.

| | |
|---------------|------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authority | The format is https://instancename.okta.com It must be the URL of your Okta instance. |
| Client ID | The Client ID you get after registered the Bold Reports application in Okta website. |
| Client Secret | The Client Secret you get after registered the Bold Reports application in Okta website. |
| Identifier | preferred_username |

Single Sign-On(SSO) with Auth0 authentication in Bold Reports

Users can be added to the Bold Reports application using the Auth0 provider. By importing them, you can share the reports and email exported reports with them.

How to register the Bold Reports application in Auth0

This section explains how to perform Single Sign-On for users in Auth0 with the Bold Reports application.

This configuration has to be done in Auth0 website.

Prerequisites

- An admin account in Auth0.
- Install Bold Reports application.

Steps to register the Bold Reports application

1. Login to the Auth0 website with an **admin** account.

![Auth0 Login Admin Account](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/auth0-login-admin-account.png)

2. Click **Applications** in the left menu and then click **CREATE APPLICATION** button.

![Auth0 create Application](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/auth0-create-application.png)

3. Click **Regular Web Applications** and proceed with Next.

Single Sign-On(SSO) with Auth0 authentication in Bold Reports
BoldReports

Enable Auth0 authentication in

![Select Regular Web Application](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/select-regular-web-application.png)

4. The application will be registered and directed to the application details page. Use the **Client ID** and **Client Secret** in the BoldReports application.

![auth0 ClientId and Client Secret](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/auth0-clientid-client-secret.png)

5. Scroll down and save the Allowed **Callback URLs**.

![Auth0 Save Callback URL](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/auth0-save-callback-url.png)

6. This URL must be the redirect URI in the settings of your BoldReports application as in the following screenshot.

![Login Redirect URI](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/login-redirect-uri.png)

7. Fill the details and save the changes.

After successful registration in Auth0, save these settings in BoldReports settings page to enable this authentication.

Enable Auth0 authentication in BoldReports

Configure the settings in Bold Reports as in the following snap to enable the authentication using Auth0.

![Configure Bold Report OpenID Auth0](/static/assets/on-premise/images/authentication/single-sign-on/openid-connect/auth0/configure-boldreport-openid-auth0.png)

The fields to be saved in the Bold Reports to enable the Auth0 for authentication is explained as follows.

| | |
|---------------|-------------------------------------------------------------------------------------------|
| Provider Name | It represents the name of the authentication provider to be displayed in the login page. |
| Provider Logo | It represents the logo of the authentication provider to be displayed in the login page. |
| Authority | The format is https://instancename.auth0.com It must be the URL of your Auth0 instance. |
| Client ID | The Client ID you get after registered the Bold Reports application in Auth0 website. |
| Client Secret | The Client Secret you get after registered the Bold Reports application in Auth0 website. |
| Identifier | email |

Office 365 for authentication

The Bold Reports On-Premise supports importing Azure Active Directory users and allows them to do Single Sign-On (federated identity) into the Bold Reports On-Premise using their credentials. You can share or email the reports with Office 365 users and groups. The Office 365 uses the cloud-based user authentication service Azure Active Directory to manage users. So, the Report Server supports the Office 365 users and groups to be imported and sign in to it.

The Report Server supports the following functionalities from the Azure Active Directory:

- [Setup Azure Active Directory](#) settings with the Bold Reports On-Premise.
- [Import users](#) from the Azure Active Directory.
- [Import groups](#) from the Active Directory.
- [Synchronize user details](#) with the Azure Active Directory.
- [Synchronize group details and its users](#) with the Azure Active Directory.
- [Schedule the synchronization of user and groups](#) between the Bold Reports On-Premise and the Azure Active Directory.

Embed in Application

Bold Reports Report Server is designed with an option of integrating the reporting functionalities in application. For that, a comprehensive set of API has been provided to have a server integration within application. There are three options for integrating reporting functionalities into your applications. Those are REST API, Reporting Component, and Embed URL. Each option provides a different approach for integrating reporting functionalities into your applications.

REST API

Bold Reports Report server is designed to manage the Report Server functionalities from the application with the help of built-in REST API. You can refer more about this from the [REST API reference](#) and the following articles,

1. [Authorization Token](#).
2. [Get list of Reports from Report server](#)
3. [Export report from Report Server](#)

Reporting Components

Embedded Reporting Tools available with Bold Reports to integrating the Reporting Viewing and Editing functionalities into your application. The component provides the capabilities to the viewer and edits the report with Report Server along with the built-in Report Service. You can refer more about this from the following sections,

- [View report through Report Viewer](#)
- [Integrating Report Designer](#)

Embed URL

URL access is another option for integrating report viewing into your applications. You have to generate the embedded URL from embedded code and that can be used with the iFrame to embed report view functionalities with your application. You can refer more about this from the following sections,

[Get Embed Code of Report](#)

Embed Code of Report

Embed code of report explains how to embed Bold Reports Server reports into other application on iframe using the embed URL.

See also

[Get Embed Code of Report](#)

Get an embed code of report

This section explains how to get an embed code of report from the Bold Reports On-Premise to use it in your web application page. Embed server reports provide the option to render the reports into other web applications. Follow these steps to embed server reports into other web applications.

Getting an embed URL

1. To get the embed URL, click the Share icon of the respective report.

![Share Icon](/static/assets/on-premise/images/embed-server-reports-in-application/iframe/share-option.png)

2. Click the Get Embed Code link in the Share dialog box.

![Disabled Comment](/static/assets/on-premise/images/embed-server-reports-in-application/iframe/get-embed-code-link.png)

3. By Default, Reports Comments and Views options are disabled. You can enable this option by sliding the button given in Embed dialog box.

![Sliding Option](/static/assets/on-premise/images/embed-server-reports-in-application/iframe/sliding-option.png)

| | |
|-----------------|----------------------------------------------------------------------------------|
| Report Comments | Report comment panel would be shown in embedded view, if enabled this parameter. |
| Views | Report views panel would be shown in embedded view, if enabled this parameter. |

Steps to embed report using the embed URL

1. Copy the embed URL by clicking the copy icon from the Embed Code dialog box.

![Embed URL](/static/assets/on-premise/images/embed-server-reports-in-application/iframe/embed-code-copy-option.png)

2. Embed the copied URL into other web application with the help of the following code snippet.

```
'html
```

```
<html>
```

Report Server report using the Bold Reporting Tools Report Viewer Steps to embed report using the embed URL

```
<head>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type">
<meta content="utf-8" http-equiv="encoding">
</head>
<body>
<div id="report">
<iframe src='http://localhost:61405/reporting/en-us/site/site1/reports/81cd3e3e-d611-4ef8-9762-8012bae416be/Sample%20Reports/Grouping%20Aggregate?isembed=true' id='report-frame' width='100%' height='720px' allowfullscreen frameborder='0'></iframe>
</div>
</body>
</html>
`
```

Run the web application, it will show the login window or report based on the type of the report. Click **login** and login with valid user credentials and the report will render automatically.

![Embed Login Page](/static/assets/on-premise/images/embed-server-reports-in-application/iframe/login.png)

![Embed Report](/static/assets/on-premise/images/embed-server-reports-in-application/iframe/report-loaded.png)

Public embedding

If you want to embed the report without user authentication, then make the report public and embed it. The public report will be rendered automatically without credentials. You cannot enable the comments panel for public report embedding.

Embedding the public report will not work when public report setting is disabled. Learn more about the report settings [here](#).

Private embedding

Private reports can be embedded, which are accessible to users in the Bold Reports, who has the read permission, and these reports would be requested to login into Bold Report server before rendering. So, embedded report will show the login page whenever private reports are embedded and the user is not authenticated yet.

By default, report access mode will be set to private until it is changed to the public by the owner of the report.

Report Server report using the Bold Reporting Tools Report Viewer

Bold Report Server provides build-in Restful API to embed Report Server reports using the Bold Reporting Tools Report Viewer. For this, set the `serviceAuthorizationToken`, `reportServiceUrl`, and `reportPath` property for Report Viewer.

`serviceAuthorizationToken` - Pass the access token of the user. Get access token using the Bold Reports Server [Authorization token API](#).

`reportServiceUrl` - Bold Reports On-Premise built-in service URL.

`reportPath` - Path of report, which is added on server.`{/category name}/{report name}`

You can also load the report using the Guid instead of report location. Set the Guid of the report in `reportPath` as like as `reportPath: '91f24bf1-e537-4488-b19f-b37f77481d00'`. You can use Bold Report Server [Get Items API](#) and find the Guid of the report.

You can get more details of embedding with Bold Reporting Tools respective platforms as follows.

[Angular](#)

[ASP.NET Core](#)

[ASP.NET MVC](#)

[ASP.NET Web Forms](#)

[Javasctipt](#)

[React](#)

[UWP](#)

[WPF](#)

Integrate a Bold Reporting Tools Report Designer

Bold Report Server provides built-in restful API to integrate Bold Reporting Tools Report Designer into other applications. For this, set the `serviceAuthorizationToken` and `serviceUrl` property for Report Designer.

`serviceAuthorizationToken` - Pass the access token of the user. Get access token using the Bold Reports Server [Authentication API](#).

`serviceUrl` - Bold Reports On-Premise built-in service URL.

You can get more details of embedding with Bold Reporting Tools respective platforms as follows.

[Angular](#)

[ASP.NET Core](#)

[ASP.NET MVC](#)

[ASP.NET Web Forms](#)

[Javasctipt](#)

Tenant Management

This topic describes about the tenant management in Bold Reports On-Premise Edition.

You can create multiple sites and maintain users of Bold Reports On-Premise.

[Create site](#)

Creating a New Site

This section briefly explains the steps involved in creating a new site in Bold Reports On-premise multi-tenant application.

Site Creation

You can create another site in Bold Reports On-premise application by clicking on **Create Site** button in site management module.

![Create Site](/static/assets/on-premise/images/tenant-management/create-site/create-site-button.png)

This step holds the site details such as name and identifier. The site identifier should be unique and it is part of your site URL.

![Site Registration](/static/assets/on-premise/images/tenant-management/create-site/site-creation.png)

Select Database

This step stores the reports, users and their access permissions in SQL Server Database or PostgreSQL Server Database. You can connect to the existing SQL Server instance with the below options.

- By creating new **Bold Reports Site** database.

![Select Database](/static/assets/on-premise/images/tenant-management/create-site/select-database.png)

- By choosing one of the database from **Select a Database** drop down for creating Bold Reports Server tables in that database.

![Select Database ExistingDb](/static/assets/on-premise/images/tenant-management/create-site/select-database-existing.png)

You can also connect to the existing PostgreSQL Server instance with the below options.

- By creating new **Bold Reports Site** database.

![Select Database](/static/assets/on-premise/images/tenant-management/create-site/select-postgresql-database.png)

- By choosing one of the database from **Select a Database** drop down for creating Bold Reports Server tables in that database.

![Select Database ExistingDb](/static/assets/on-premise/images/tenant-management/create-site/select-postgre-database-existing.png)

Note: The credentials that is given to connect to the SQL Server or PostgreSQL Server instance must have permissions to

- Create Database
- Create Table
- Insert
- Update Table
- Alter Table
- Select

- Drop Table
- Drop Database

Select Storage

User can select the preferred storage type File Storage or Blob Storage to store the report,datasource and dataset in BoldReports Server

![SQL Server DataStore](/static/assets/on-premise/images/tenant-management/create-site/select-storage-type.png)

Blob Storage

User can select the Blob Storage by giving Azure Blob Credential in the required field

![Blob Storage](/static/assets/on-premise/images/tenant-management/create-site/create-storage-type-blobstorage.png)

Select Administrator

This step used to select the user to provide complete control over the site.

The selected users have the following permissions,

- Create Reports
- Create Data Sources
- Create Schedules
- Create Datasets
- Create Users
- Create Groups
- Manage Permissions for users and groups

After selecting the user, proceed with **Create and launch site**.

![Select Administrator](/static/assets/on-premise/images/tenant-management/create-site/select-administrator.png)

Localization

Localization is the process of adapting a website into different linguistic and cultural contexts - involving much more than the simple translation of text.

The default language is English “en-US” for Bold Reports On-Premise. Read the following documentation for more details.

Adding new localizations

Clone your own culture for the [Bold Report Server](#) and add it in the application anytime.

![clone-po-file](/static/assets/on-premise/images/localization/clone-po-file.png)

Adding localizations to the Bold Report On premise

- Copy your own culture **messages.po** file from the location where you are cloned from GitHub

![Copy-Po-File](/static/assets/on-premise/images/localization/copy-po-file.png)

- Create a folder in **C:\Bold Reports\Report Server\locale** with {language code}-{country code} and paste the cloned messages.po file inside the newly created folder.

For example, if you are translating to Germany, create a folder named de-DE and paste the messages.po like the below.

C:\Bold Reports\Report Server\locale\de-DE\messages.po

![Create New Folder](/static/assets/on-premise/images/localization/create-new-folder.png)

- Then refresh Your Bold Report On Premise source and select the preferred language.

![Select Language](/static/assets/on-premise/images/localization/select-language.png)

User Profile

This section describes the options available for users to view and manage their profile. The options are:

- View profile
- Edit profile
- View permissions
- Change password
- Edit notification settings

View profile

To view your profile details, click the **Profile** icon in the Report Server left-side panel.

![View user profile details](/static/assets/on-premise/images/user-profile/view-user-profile.png)

Edit profile

User can edit their profile details in Bold Reports account. Click **Bold Reports Account** in profile page.

![Bold Reports Account](/static/assets/on-premise/images/user-profile/bold-reports-account.png)

You can edit and change the first name, last name, phone number, and profile picture of your account.

![Edit user profile options](/static/assets/on-premise/images/user-profile/edit-user-profile.png)

Change password

To change the user password to log in to the Report Server, click **Change Password**.

![Change password button](/static/assets/on-premise/images/user-profile/change-password-button.png)

Set new password, and then click **Save**.

![Change user login password](/static/assets/on-premise/images/user-profile/edit-user-password.png)

My permissions

You can view your access permission list for each resources like reports, data sources, files, and schedules in the Report Server.

![View user access permissions](/static/assets/on-premise/images/user-profile/view-permission.png)

Edit notification settings

You can view and edit the following notification settings in the Report Server:

- System notifications
- Mail notifications
- Auto watch for comments of created items.
- Auto watch for comments of accessible items.

![View and edit notification settings](/static/assets/on-premise/images/user-profile/edit-notification-settings.png)

Custom rebranding

You can rebrand the Report Server by changing the organization name, site URL, login screen logo, welcome text, header logo, favorite icon, email logo, favicon, time zone, and date time format.

To rebrand the Report Server, click the Settings option in the Report Server left panel and navigate to the **Site** tab as shown in following image.

![Site Settings option](/static/assets/on-premise/images/custom-rebranding/site-settings.png)

Organization name

Organization name is the name displayed on the title bar of the browser. To modify the browser title, you can change the **Organization name** field of the Report Server.

Site URL

Modify the **Site URL** field to change the Report Server URL in the Server Settings page and to get this change reflected, you should configure the same in the IIS. Refer to this [link](#) to add binding in IIS and change site URL.

Login screen

You can change the following items in the login screen.

- For the login page logo, the preferred image size is **200x40** pixels. By default, the BoldReports logo is displayed.
- In welcome note, the maximum characters allowed is 70. Default welcome text is "Welcome to Bold Reports On-Premise".

Header

You can change the header logo image, and the preferred image size is **40x40** pixels. The Report Server will have BoldReports logo as default main screen logo.

Favicon

You can change favicon, and the preferred image size is **40x40** pixels. By default, the Report Server will have BoldReports favicon.

Email

You can change email logo, and the preferred image size is **40x40** pixels. By default, the Report Server will have BoldReports email logo.

Display

Time zone

By default, the Report Server sets the time zone of the system where it is installed. You can change the time zone for the Report Server based on the requirement.

Date format

Report Server will have **MM/dd/yyyy** as the default date format. You can modify the date format based on your culture.

Time format

You can change the time format of the Report Server, and the default time format is 12 hour.

Powered by Syncfusion

Allows you to show or hide **Powered by Syncfusion** in the footer of the Report Server. By default, this will be shown.

Copyright information

Allows you to show or hide **Copyright Information** in the footer of the Report Server. By default, this will be shown.

Report Designer

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

Key features

Data sources --- Supports commonly used data sources such as Microsoft SQL Server, SQL CE, Oracle, XML, OLEDB, ODBC, SASS, Web API, MySQL, and PostgreSQL.

Built-in query builder --- Provides a convenient user interface that helps both technical and non-technical users to create and view relationships between tables.

Report items --- Provides a rich set of report items like text box, image, line, rectangle, table, list, matrix, more than 20 chart types, sub reports, and barcodes for clear visualization of your data.

Interactive design surface --- Easily arrange report items on the Report Designer surface using simple drag-and-drop operations. Grid lines and snap-to-grid options simplify positioning and aligning of report items.

Report parameters --- Supports creating report parameters to specify the data to filter in a report, connect related reports together, and vary report presentation.

Image manager --- Allows adding and managing embedded images in a report. Embedding an image ensures the image is always available to a report.

Property panel --- Allows customizing the appearance of report items and editing their property values. Users can specify static values or expressions as values in the report item properties.

Interactive reports --- Supports creating a report with extremely powerful interactive features provided only by RDL reports, such as drill through and drill down to display data and information in detail.

Manipulate your data --- Allows manipulating data with a rich set of features such as sorting, filtering, grouping, and calculated fields.

Built-in expression editor --- Allows you to create simple and complex RDL reporting expressions. Supports the use of constants, built-in fields, parameters, data sets, variables, operators, common functions, etc.

Embedded custom code --- Supports embedding Visual Basic and .NET code directly within report layouts.

Report Creation

This section describes simple steps to design a report using Web Report Designer.

![Report designer initial view](/static/assets/on-premise/images/report-designer/report-creation/report-designer-initial-view.png)

Create Data

1. To add a data, click on the **Data** icon in the **Data Configuration** panel. It opens the **Data** panel.

![Data panel icon](/static/assets/on-premise/images/report-designer/report-creation/data-icon-configuration-panel.png)

2. Click on **New Data** button in the data panel.

![New data button](/static/assets/on-premise/images/report-designer/report-creation/new-data-button-data-panel.png)

3. Choose **SQL** type to connect data.

![Data source types panel](/static/assets/on-premise/images/report-designer/report-creation/datasource-types-panel.png)

4. In the new data source panel,

- In **Name** field, specify the data source name without special characters.
- In **Server Name** field, you need to select existing server in the local network from the drop-down list or specify the specific remote server name like `myserver.domain.com`.
- In **Authentication Type** field, choose **Windows** or **SQL Server** authentication.
- If you have chosen **SQL Server** authentication, specify the **username** and **password** of the server.
- In **Database** field, choose or enter a existing valid database e.g. AdventureWorks.

![Data source fields panel](/static/assets/on-premise/images/report-designer/report-creation/datasource-fields-panel.png)

5. Click the **Connect** button. Now the following view will be displayed.

![Query designer initial view](/static/assets/on-premise/images/report-designer/report-creation/query-designer-full-view.png)

Here, an **AdventureWorks** database is used for demonstration.

Edit dataset name

You can edit the name of the **Data** in the **Name** field that is available in toolbar pane.

![Dataset name field](/static/assets/on-premise/images/report-designer/report-creation/edit-dataset-name.png)

Drag and drop table in query designer

The left pane holds the **tables**, **views**, and **procedures** associated with the connected database. Drag your preferred table or view from the left pane and drop into the center pane labeled with **Drag and Drop table here** like below:

![Drag and drop query table to design area](/static/assets/on-premise/images/report-designer/report-creation/drag-drop-table-in-query-designer.png)

Now, the table will be dropped in the design area like below.

![Query designer with query table](/static/assets/on-premise/images/report-designer/report-creation/drag-drop-table-in-query-designer-output.png)

Execute query

1. You can execute and visualize the data by using **Run** option in tools pane.

![Run query icon](/static/assets/on-premise/images/report-designer/report-creation/run-query-in-querydesigner.png)

2. Now, the data will be retrieved based on the specified query.

![Execute query output](/static/assets/on-premise/images/report-designer/report-creation/execute-query-output.png)

Save Data

1. Click on the **Finish** button in the tools pane.

![Finish data button](/static/assets/on-premise/images/report-designer/report-creation/finish-data-connection.png)

2. Your dataset should now be listed in the **Data** panel like below.

![Data list view](/static/assets/on-premise/images/report-designer/report-creation/data-list-view.png)

3. Expand the icon to view the data fields.

![Data fields list](/static/assets/on-premise/images/report-designer/report-creation/data-fields-list.png)

Add table report item

The left pane in the design view consists of basic items, data region, data visualization, and sub reports to design an interactive report.

Here, the **Table** report item is used for demonstration.

1. Select the **Table** item under **Data Regions** in the item panel, then drag and drop it to the design area.

![Table report item](/static/assets/on-premise/images/report-designer/report-creation/drag-drop-table-item.png)

2. The above action will render the **Table** with two rows and three columns in the design area.

![Table initial view](/static/assets/on-premise/images/report-designer/report-creation/table-initial-view.png)

Assign data

This step is applicable only for the report items that belongs to **data visualization** and **data region** category.

Assign the dataset to the **Dataset** property of the table.

![Assign dataset to the table](/static/assets/on-premise/images/report-designer/report-creation/assign-dataset-to-table.png)

Add column header

1. Select the first cell in the table, and enter the column header text as **ProductID** to the **Content** property of table.

![Enter text in table cell](/static/assets/on-premise/images/report-designer/report-creation/enter-text-in-table-cell.png)

2. Similarly, you can add required column header text to other cells in the table.

![Assign column text output](/static/assets/on-premise/images/report-designer/report-creation/assign-column-text-output.png)

Assign data fields in table cell

1. Select the table cell and click on the **Data assign** menu icon to open data assign menu.

![Open data assign menu](/static/assets/on-premise/images/report-designer/report-creation/open-data-assign-menu.png)

2. Assign the **ProductID** field in the table cell.

![Assign fields to table cell](/static/assets/on-premise/images/report-designer/report-creation/assign-fields-to-table-cell.png)

3. Similarly, you can assign the required data fields to the table cell.

![Output of assign fields procedure](/static/assets/on-premise/images/report-designer/report-creation/assign-fields-to-table-cell-output.png)

4. Assign expression to the table cell.

![Assign expression to table cell](/static/assets/on-premise/images/report-designer/report-creation/add-expression-to-table-cell.png)

5. In the expression dialog, add the following expression
=Fields!OrderQty.Value*Round(Fields!UnitPrice.Value,2) and click OK.

![Assign expression to table cell](/static/assets/on-premise/images/report-designer/report-creation/unit-price-expression.png)

6. Now, the table will look like below,

![Assign expression to table cell](/static/assets/on-premise/images/report-designer/report-creation/expression-output-in-table-cell.png)

Resize the column

To improve the report readability, we can resize the table row height and column width.

1. Place the mouse pointer in the respective column border.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-creation/resize-the-table-column.png)

2. Drag the column gripper horizontally, to adjust the column width.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-creation/adjust-column-width-output.png)

Resize the row

1. Place the mouse pointer in the respective row border.

![Resize the table row](/static/assets/on-premise/images/report-designer/report-creation/resize-the-table-row.png)

2. Drag the row gripper vertically, to adjust the row height.

![Adjust row height of the table](/static/assets/on-premise/images/report-designer/report-creation/adjust-row-height-output.png)

Customize the appearance

Open the PROPERTIES pane. This pane holds some general settings and some specific to the report item. Configure the desired settings to the table for better report design and to improve report readability.

![Customize table cell properties](/static/assets/on-premise/images/report-designer/report-creation/customize-table-cell-properties.png)

[Add total](#)

1. Select the second row, first cell and Right click -> **Add Total**

![Add total menu](/static/assets/on-premise/images/report-designer/report-creation/add-total-menu.png)

2. Now, select the third row, first four columns and Right click -> **Merge cells**

![Merge cells in table](/static/assets/on-premise/images/report-designer/report-creation/merge-cells-in-table.png)

3. Modify the cell content as **Total** and align the content to **Right** side.

![Modify the cell content](/static/assets/on-premise/images/report-designer/report-creation/modify-total-cell-content.png)

4. Select the third row, last cell and open the **Data Assign** menu. Then, click on **Add Expression**.

![Add expression menu](/static/assets/on-premise/images/report-designer/report-creation/add-expression-menu.png)

5. In the expression dialog, add the following expression

=Sum(Fields!OrderQty.Value*Round(Fields!UnitPrice.Value,2)) and click OK.

![Provide valid expression](/static/assets/on-premise/images/report-designer/report-creation/round-of-the-line-total.png)

Now, the table design will look like below.

![Final design view](/static/assets/on-premise/images/report-designer/report-creation/final-design-view.png)

[Save report](#)

Once you are done with the report designing, to save a report refer [How to save report](#) section.

[Preview report](#)

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-creation/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Table report preview](/static/assets/on-premise/images/report-designer/report-creation/report-preview-page-1.png)

See also

[Design a Basic Chart Report](#)

[Create an Embedded DataSource](#)

[Create an Embedded DataSet](#)

[Link a Shared DataSource into a Report](#)

[Link a Shared DataSet into a Report](#)

[Create a Duplicate Copy of DataSource in a Report](#)

[Create a Duplicate Copy of DataSet in a Report](#)

[Add a Report Parameter to a Report](#)

[Embed an Image in a Report](#)

Manage data

A report must have a data source and dataset to include data. Bold Report Designer provides support to manage your data in reports without writing single line of code. You can explore your data from any of the local or server data bases. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

See also

[Create Embedded Datasource](#)

[Link Shared Datasource](#)

[Edit Embedded Datasource](#)

[Edit Shared Datasource](#)

[Delete Datasource](#)

[Duplicate Datasource](#)

[Create Embedded Dataset](#)

[Link Shared Dataset](#)

[Edit Embedded Dataset](#)

[Edit Shared Dataset](#)

[Delete Dataset](#)

[Duplicate Dataset](#)

Data source

Bold Report Designer provides support to create an embedded or shared data source using built-in or data processing extension. You can have any number of data sources that connect to different local or server data bases. It provides connection to major data providers such as **Microsoft SQL Server**, **SQL CE**, **Oracle**, **XML**, **OLEDB**, **ODBC**, **SASS**, **Web API**, **MySQL**, and **PostgreSQL** for exploring data and designing reports with a wide range of data sources.

Embedded data source

Allows you to create an embedded data source with your own data connection. The embedded data source is specific to a report and you can use the connection string as expressions.

Shared data source

You can add a shared data source to report that is stored on a Report Server and shared by multiple reports. The shared data source provides better security by hiding the connection details.

Custom data processing extensions

Provides support to add any additional data providers to connect to your own database or extend functionalities of the existing data providers. The .NET framework compatible custom data processing extensions need to be created using data extension interface. With minimal configurations, you can use any additional data source such as SASS, WebApi, MySQL, PostgreSQL, MongoDB, etc.

See also

[Create Embedded Datasource](#)

[Link Shared Datasource](#)

[Edit Embedded Datasource](#)

[Edit Shared Datasource](#)

[Delete Datasource](#)

[Duplicate Datasource](#)

Create an embedded data source

This section guides you to create an embedded data source in Bold Report Designer. The embedded data source can only be used by the report in which it is embedded.

Add a new data source

To bind data to a report item, a minimum of one data source is needed. A data source can be created through the below steps:

1. Click on the **Data** icon in the configuration panel.

![Data icon configuration panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-icon-configuration-panel.png)

2. In the **Data** panel, click on the **switcher** icon on the top-right corner.

![Data panel switcher icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-panel-switcher-icon.png)

3. Click the **Datasources** in context menu, to open **DATASOURCES** panel.

![New data source panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/new-data-source-panel.PNG)

4. In the **DATASOURCES** configuration panel, click on the **NEW DATASOURCE** button.

5. In the connection type panel, choose the data source type that you want to connect.

Here, **SQL** connection type is used for demonstration.

![Connection types panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/connection-types-panel.png)

6. In the new connection panel,

- In **Name** field, specify the data source name without special characters.
- In **Server Name** field, you need to select existing server in the local network from the drop-down list or specify the specific remote server name like **myserver.domain.com**.
- In **Authentication Type** field, choose **Windows** or **SQL Server** authentication.
- If you have chosen **SQL Server** authentication, specify the **username** and **password** of the server.
- In **Database** field, choose or enter an existing valid database e.g. AdventureWorks.

![New connection panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/save-new-data-source.png)

7. Now, click on the **Save** button and the new embedded data source will be added in data source list pane like below.

![Data source list view](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-source-list-view.png)

8. The embedded data source is created in the report and it is ready to create/use the data.

Modify an embedded data source

This section guides you to modify an embedded data source in Bold Report Designer. You can edit a data source through the following steps:

1. Select a data source in the **DATASOURCES** panel that you want to edit.

![Data source item menu icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-source-item-menu-icon.png)

2. Click the above highlighted icon to open the **Edit** context menu of the selected data source.

![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-panel-context-menu.png)

3. Choose **Edit** option from the context menu, to open **Edit Connection** panel.

![Edit connection panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/edit-connection-panel.png)

4. After modifying the fields, click on **Save** button.

Link a shared datasource

This section guides you to link a shared datasource in Bold Report Designer.

1. Click the **Data** icon in the configuration panel to launch a **Data** configuration.

![Data icon configuration panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-icon-configuration-panel.png)

2. In the **Data** panel, click on the **switcher** icon on the top-right corner.

![Data panel switcher icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-panel-switcher-icon.png)

3. In the context menu, click **Datasources** to launch the **DATASOURCES** configuration panel.

![New data source panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/new-data-source-panel.PNG)

4. In the **DATASOURCES** configuration panel, click on the **NEW DATASOURCE** button. In the connection type panel, choose the **Shared** data source type.

![Connection types shared data](/static/assets/on-premise/images/report-designer/manage-data/datasource/connection-types-shared-data.png)

5. After choosing the **Shared** datasource type, the new connection wizard will be shown as below:

![Shared data source fields](/static/assets/on-premise/images/report-designer/manage-data/datasource/shared-data-source-fields.png)

6. In **Name** field, specify the data source name without special characters.
7. In **Shared DataSource** field, you can select an existing shared data sources in the server from the drop-down list.

![New connection panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/save-new-shared-data-source.png)

8. Now, click on the **Save** button and the new shared data source will be added in data list pane like below.

![Data source list view](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-source-list-view.png)

Note: You can link multiple shared data sources into the report.

Modify shared datasource

This section guides you to modify an shared data source connection in Bold Report Designer. You can edit a shared data source through the following steps:

1. Select a data source in the **DATASOURCES** panel that you want to edit.
![Data source item menu icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-source-item-icon.png)
2. Click on the above highlighted icon to open the **Edit** context menu of the selected data source.
![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-panel-context-menu.png)
3. Choose **Edit** option from the context menu, to open **Edit Connection** panel.
![Edit connection panel](/static/assets/on-premise/images/report-designer/manage-data/datasource/edit-shared-data-source-panel.png)
4. After modifying the fields, click on **Save** button.

Delete a data source

This section guides you to delete an embedded data source from the report in Bold Report Designer. You can delete a data source through the following steps:

1. Select a data source in the **DATASOURCES** panel that you want to delete.
![Data source item menu icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-source-item-icon.png)
2. Click the above highlighted icon to open the delete context menu for the selected data source.
![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-panel-context-menu.png)
3. Choose **Delete** from the context menu, it will launch the confirmation dialog like below.
![Delete data source confirmation](/static/assets/on-premise/images/report-designer/manage-data/datasource/delete-data-source-confirmation.png)
4. Click on the **Yes** button to remove the datasource from the report.

Duplicate a data source

This section guides you to duplicate an existing data source connection in Bold Report Designer. It can be done through the following procedure:

1. Select the data source that you need to duplicate in the **DATASOURCES** panel.

![Data source item menu icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-source-item-menu-icon.png)

2. Click the above highlighted icon to open the context menu.

![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/datasource/data-panel-context-menu.png)

3. Choose **Clone** option from the context menu, to duplicate the selected data source.

![Image represents the duplicate datasource icon](/static/assets/on-premise/images/report-designer/manage-data/datasource/clone-data-source-option.png)

4. Now a duplicated data source will be created like below.

![Image represents the datasource is duplicated](/static/assets/on-premise/images/report-designer/manage-data/datasource/duplicate-data-source-represenation.png)

Report data sets

A dataset contains a query and collection of fields to represent the data returned from the data source.

Embedded data set

You can create a data set using an embedded or shared data source. It stores query used to process the report data. The embedded data set is specific to a report and you can add query parameters, filters, table relationships, and have the query string as expression.

Shared data set

Like shared data source, you can add a dataset that is published on a Report Server. It provides an easy way to change query at one place and reflect it in referred reports.

See also

[Create Embedded Dataset](#)

[Link Shared Dataset](#)

[Edit Embedded Dataset](#)

[Edit Shared Dataset](#)

[Delete Dataset](#)

[Duplicate Dataset](#)

Create an embedded dataset

This section guides you to create an embedded dataset in Bold Report Designer.

Design query data

1. Click the **Data** icon in the configuration panel to launch a **Data** configuration.

![Data icon configuration panel](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-icon-configuration-panel.png)

2. Click on the **New Data** button in **Data** panel.

![New data button](/static/assets/on-premise/images/report-designer/manage-data/dataset/new-data-button.png)

3. In the connection type panel, click the data source type that you want to connect. Here, **SQL** connection type is used to demonstrate.

![Connection types panel](/static/assets/on-premise/images/report-designer/manage-data/dataset/connection-types-panel.png)

4. In the new data source configuration panel, fill the server name and related details.

![New data source connection panel](/static/assets/on-premise/images/report-designer/manage-data/dataset/new-data-source-connection-panel.png)

5. Click on the **Connect** button, now the following view will be displayed.

![Query designer full view](/static/assets/on-premise/images/report-designer/manage-data/dataset/query-designer-full-view.png)

6. You can edit the name of the **Data** in the **Name** field that is available in toolbar pane.

![Edit dataset name](/static/assets/on-premise/images/report-designer/manage-data/dataset/edit-dataset-name.png)

7. You can drag the tables and views associated with the connected database from left pane.

Dragged table or view from the left pane can be dropped into the center pane labeled with **Drag and Drop tables here** like below.

![Drag and drop query table to design area](/static/assets/on-premise/images/report-designer/manage-data/dataset/drag-drop-table-in-query-designer.png)

After you drop the item into the center pane, it displays like below:

![Query designer with query table](/static/assets/on-premise/images/report-designer/manage-data/dataset/drag-drop-table-in-query-designer-output.png)

8. To search the table or view from the data collection, you can use the **Search** field in the left pane of the query designer.

![Query designer table search pane](/static/assets/on-premise/images/report-designer/manage-data/dataset/query-designer-table-search-pane.png)

Save Data

1. Click on the **Finish** button in the tools pane.

![Finish data button](/static/assets/on-premise/images/report-designer/manage-data/dataset/finish-data-connection.png)

2. Your dataset should now be listed in the **Data** panel like below.

![Data list view](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-list-view.png)

3. Expand the icon to view the data fields.

![Data fields list](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-fields-list.png)

Modify an embedded dataset

This section guides you to modify an embedded dataset in Bold Report Designer. You can edit a data source through the following steps:

1. Select a data in the **DATA** panel that you want to edit.

![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-item-menu-icon.png)

2. Click on the above highlighted icon to open the **Edit** context menu of the selected data.

![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-panel-context-menu.png)

3. Choose **Edit** option from the context menu. Now the below view will be displayed to handle your modification.

![Query designer edit view](/static/assets/on-premise/images/report-designer/manage-data/dataset/query-designer-edit-view.png)

4. After modifying the fields, click **Finish** button.

Link a shared dataset

You can create a shared dataset in the report server and it can be linked with the reports in Report Designer. This section guides you to create a shared dataset in Bold Report Designer.

1. Click the **Data** icon in the configuration panel to launch a **Data** configuration.

![Data icon configuration panel](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-icon-configuration-panel.png)

2. Click on **SHARED DATA** button in **Data** panel.

![Click on shared data button](/static/assets/on-premise/images/report-designer/manage-data/dataset/shared-data-button.png)

3. After clicking on the **SHARED DATA**, the new dataset wizard will be shown as below:

![Click on new dataset button](/static/assets/on-premise/images/report-designer/manage-data/dataset/new-dataset-button.png)

4. In **Name** field, specify the dataset name without special characters.
5. In **Shared DataSet** field, you can select existing shared datasets in the server from the drop-down list.

![Shared data fields panel](/static/assets/on-premise/images/report-designer/manage-data/dataset/shared-data-fields.png)

6. Click on the **Save** button. Now, a new shared dataset will be added in the report like below.

![Data list view](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-list-view.png)

Note: You can link multiple shared dataset into the report.

Modify shared dataset

This section guides you to modify an shared data connection in Bold Report Designer. You can edit a shared data through the following steps:

1. Select a data in the **DATA** panel that you want to edit.

![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-item-menu-icon.png)

2. Click on the above highlighted icon to open the **Edit** context menu of the selected data.

![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-panel-context-menu.png)

3. Choose **Edit** option from the context menu. Now, the below view will be displayed to handle your modification.

![Query designer edit view](/static/assets/on-premise/images/report-designer/manage-data/dataset/edit-shared-data-panel.png)

4. After modifying the fields, click on **Save** button.

Delete a dataset

This section guides you to delete an embedded data from the report in Bold Report Designer. You can delete a data through the following steps:

1. Select a data in the **DATA** panel that you want to delete.
![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-item-menu-icon.png)

2. Click the above highlighted icon to open the delete context menu for the selected data.
![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-panel-context-menu.png)

3. Choose **Delete** from the context menu, it will launch the confirmation dialog like below.
![Delete data confirmation](/static/assets/on-premise/images/report-designer/manage-data/dataset/delete-data-confirmation.png)

4. Click on the **Yes** button to remove the data from the report data list.

Duplicate a dataset

This section guides you to duplicate an existing dataset connection in Bold Report Designer. It can be done through the following procedure:

1. Select the dataset that you need to duplicate in the **DATA** panel.
![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-item-menu-icon.png)

2. Click the above highlighted icon to open the context menu.
![Data panel context menu](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-panel-context-menu.png)

3. Choose **Clone** option from the context menu, to duplicate the selected dataset.
![Image represents the duplicate dataset icon](/static/assets/on-premise/images/report-designer/manage-data/dataset/clone-data-menu.png)

4. Now a duplicated data dataset will be created like below.

![Image represents the dataset is duplicated](/static/assets/on-premise/images/report-designer/manage-data/dataset/clone-data-representation.png)

Fields

Fields options can be used to create a new field when a required information is not directly available in the retrieved dataset. It will also allows you to change the column names of the existing fields in the dataset.

Refer [Create Data](#) section and create a data in report designer.

Create calculated field

1. Select a data in the **DATA** panel to which you want to add fields.

![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/data-item-menu-icon.png)

2. Click on the above highlighted icon, the context menu will open with list of options.

![Data context menu](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/context-menu.png)

3. Then, click on the **Fields...** option in the menu. Now, the available fields in the selected dataset will be listed in the **Fields** dialog as shown below.

![Open field dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/fields-dialog.png)

4. To create a new calculated field, click on the **Add** button and choose **Calculated Field** in the menu.

![Add calculated field](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/type-of-fields-in-menu.png)

5. In the **Field Name** text box, type the unique name for the field.

![Calculated field name](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/calculated-field-name.png)

6. In the **Field Source** text box, enter the static value directly or set an [Expression](#) for the field. Here, `=Month(FIELDS!OrderYear.Value)` expression is assigned to the **OrderMonth** field.

![Set value for calculated fields](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/set-value-for-calculated-field.png)

7. Now, click on the **OK** button and the newly added field will be listed under the respective dataset.

![Field updated under dataset](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/add-field-list-view.png)

The expression for a calculated field cannot contain aggregates and the field name must be unique in the dataset.

Set expression

Follow steps 1 - 5, to create calculated field.

1. To edit/create an expression of the calculated field, click on the square icon and select Expression.

![expression-icon-shown](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/expression-menu.png)

2. Expression Builder will be open as shown below.

![open-expression-dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/expression-builder.png)

To learn more about handling expressions in report designer refer [Expressions](#) section.

3. Provide the required expression and click on OK button.

![open-expression-dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/set-expression.png)

4. The icon will be indicated in Black color, if the expression is applied to the dataset field.

![set-expression](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/set-expression-indication.png)

Reset Expression

1. To reset an expression, click on the square icon and select Reset.

![Select filter reset expression](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/reset-expression.png)

2. The icon will be indicated in White color, after reset action.

![After reset expression](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/reset-expression-indication.png)

Create query field

1. Select a data in the DATA panel to which you want to add fields.

| | |
|--------|--------------|
| Fields | Delete field |
|--------|--------------|

![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/data-item-menu-icon.png)

- Click on the above highlighted icon, the context menu will open with list of options.

![Data context menu](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/context-menu.png)

- Then, click on the **Fields...** option in the menu. Now, the available fields in the selected dataset will be listed in the **Fields** dialog as shown below.

![Open field dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/fields-dialog.png)

- To create a new query field, click on the **Add** button and choose **Query Field** in the menu.

![Add query field](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/add-query-field.png)

- In the **Field Name** text box, type the unique name for the field.

![query field name](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/query-field-name.png)

- In the **Field Source** text box, enter the name of an existing field on the data set.

![query field name](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/query-field-source.png)

- Now, click on the **OK** button and the newly added field will be listed under the respective dataset.

![query field name](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/add-query-field-list-view.png)

Delete field

Click on the **Delete** icon in the right corner to remove the required fields from the dataset.

![Remove fields](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/delete-a-field.png)

Change field name

When you open the fields dialog, the available fields in the respective dataset will be listed in the **Fields** dialog as shown below.

![Field dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/fields-dialog.png)

To change the existing field name, modify the name in first textbox of the respective row and click on the **OK** button.

![Field dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/change-column-name.png)

Now, the respective field name will be updated in the dataset collection and in the list view.

![Field dialog](/static/assets/on-premise/images/report-designer/manage-data/add-dataset-field/change-column-name-list-view.png)

Add a filter to a dataset

This section guides you to add a filter at dataset level in Bold Report Designer.

Create Data

Refer [Create Data](#) section to create a data in report designer.

Apply filters at dataset level

1. Select a data in the **DATA** panel to which you want to apply filter.

![Data item menu icon](/static/assets/on-premise/images/report-designer/manage-data/dataset/data-item-menu-icon.png)

2. Click the above highlighted icon to open the filter menu for the selected data.

![Dataset filter menu](/static/assets/on-premise/images/report-designer/manage-data/dataset/filter-context-menu.png)

3. Click on **Filter...** option in the context menu, it will launch the filter dialog like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/manage-data/dataset/filters-dialog.png)

Add filters

Refer [Add filters](#) section to create filter equation.

Set expression

Refer [Set expression](#) section to set expression in filter equation.

Remove filter

Refer [Remove filter](#) section to delete a filter equation.

Example

We can filter the **ProductID** field values at dataset level like below.

1. Create a filter equation as in the below snap, and click **OK**.

![Filter equation for product id field](/static/assets/on-premise/images/report-designer/manage-data/dataset/filter-product-id-field.png)

2. Now, assign the dataset to the **Table** report item. Refer [Design a basic table report](#) section.
3. Preview the report. Observe the below snap, it displays the records for **ProductID** value **712** based on the filter equation.

![Preview output for dataset filters](/static/assets/on-premise/images/report-designer/manage-data/dataset/dataset-filter-preview.png)

Parameters

Parameters option can be used to create a new user defined parameter, edit values of existing parameters and delete parameters including query parameters that link to report parameters.

Open dataset parameters dialog

Select a data in the **DATA** panel to which you want to add, edit or delete parameters.

![Dataset panel](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/dataset-list.png)

Refer [Create Data](#) section and create a data in report designer.

Click on the above highlighted icon, the context menu will open with list of options.

![Dataset panel](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/context-menu.png)

Then, click on the **Parameters...** option in the menu to open **Parameters** dialog.

![Parameters dialog](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/report-parameters-dialog.png)

When you add a query variable to parameterize a dataset using the **@** symbol in the query when creating a dataset,

![Define parameter in query](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/add-query-parameter.png)

A **Report Parameter** will be created automatically and it will be listed in the parameters panel and dataset parameters dialog.

![Report parameter](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/parameter-panel-list.png)

Open **Parameters** dialog, the available parameters in the query will be listed in the parameters dialog like below,

![Report parameter](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/edit-parameters.png)

If you import a shared dataset into the report, the list of available parameters in the shared dataset will be listed in the parameters dialog. You can modify the values for the parameters within the report scope, it does not affect the actual query in the shared dataset.

Create parameter

1. To create a new parameter, click on the **ADD** button.

![Add parameter](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/report-parameters-dialog.png)

2. **Parameter Name** : In the **Parameter Name** text box, provide the name for the parameter. The parameter name must be unique within the dataset.
3. **Value** : In **Value** field, provide the value for the parameter. The **Value** of a parameter can be a static value or an expression, but it cannot refer to any report items or fields.

![Create parameter](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/create-parameter.png)

The parameters which you create here will act as a **User Defined** parameter in the report, neither creates a query parameter nor affects existing query parameter.

[Edit parameter](#)

Open **Parameters** dialog, the available parameters in the query will be listed in the parameters dialog like below,

![Report parameter](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/edit-parameters.png)

You can edit the **Value** for parameter and click **OK**. By default, **Value** contains an expression that refers to a report parameter.

You cannot edit the **Name** of the existing query parameter in the dataset using parameters dialog.

[Delete parameter](#)

Click on the **Delete** icon in the right corner to remove the required parameters from the dataset.

![Remove parameters](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/delete-a-parameter.png)

If you delete a parameter from the dataset, it will not be automatically removed from **Report Parameter**. You need to remove it manually if needed.

[Reorder parameters](#)

When executing a query or processing the report, the result will be retrieved from database based on the order of the parameter in the query.

The below snap depicts the parameter list before reordering the parameters.

![Before reordering the parameter](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/before-reordering.png)

To change the order of the dataset parameters after creating it, follow the below steps.

1. Click and hold the icon in the left corner of **Parameter Name** field.

![Gripper icon to drag fields](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/gripper-icon-to-perform-drag-action.png)

2. Then drag and move the dataset parameter field to higher or lower position in the list.

![Drag parameter field](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/drag-start-action.png)

3. Drop the dataset parameter field in desired position.

![Drag action demo](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/drag-action-demo.png)

The below snap depicts the parameter list after reordering the parameters.

![After reorder action](/static/assets/on-premise/images/report-designer/manage-data/dataset-parameters/after-reorder-action.png)

Transforming data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)

[Formatting columns](#)

[Query filter](#)

[Data set parameter](#)

[Query parameter](#)

[Configure expression columns](#)

Join Tables

Joining of tables is required when the dataset query design demands for more than one table. The following sections describes the steps required to create and edit the relationships between tables.

Refer [Create Data](#) section for better understanding with the following steps.

Open query joiner dialog

Click on the **Join** icon in the query designer toolbar to open **Query Joiner** dialog.

The **Join** icon in the query designer toolbar will be in **disabled** state, if there is only one table found dropped in table design view like below:

![Query joiner initial icon state](/static/assets/on-premise/images/report-designer/transforming-data/join-table/query-joiner-icon-initial-state.png)

The **Join** icon will get **enabled** once the second table found dropped in table design like below:

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/join-table/query-joiner-icon-enabled-state.png)

Create a table relation

1. Drag a table or view from the left pane and drop it into the center pane labeled with **Drag and Drop tables here** like below.

![Drag and drop a table](/static/assets/on-premise/images/report-designer/transforming-data/join-table/manual-joining-first-table.png)

2. Drag and drop another table into the design area.

![Drag and drop second table](/static/assets/on-premise/images/report-designer/transforming-data/join-table/manual-joining-second-table.png)

3. If the subsequent table being dropped, has any of its column as foreign key in any of the already dropped tables, the joining will take place automatically. Else, it will prompt the join editor like below to let you define the keys (columns) to join between this table and any one of the already dropped tables.

![Query joiner dialog](/static/assets/on-premise/images/report-designer/transforming-data/join-table/manual-joining-query-joiner-dialog-intial-view.png)

Table drop-down list

- **Left Table** - The **Left Table** drop-down list illustrates the list of table dropped in design area.

![Left table drop-down](/static/assets/on-premise/images/report-designer/transforming-data/join-table/left-table-list.png)

- **Right Table** - The **Right Table** drop-down list illustrates the list of table dropped in design area. By default, the table which you have dropped recently will be selected in this field.

![Right table drop-down](/static/assets/on-premise/images/report-designer/transforming-data/join-table/right-table-list.png)

In the below snap, the new relationship is initiated between **PurchaseOrderDetail** and **SalesOrderDetail** tables.

![Choose tables](/static/assets/on-premise/images/report-designer/transforming-data/join-table/new-table-relation.png)

Join Types

The joins are used to retrieve data from two or more data tables, based on a join condition.

Types:

1. Inner Join
2. Left Outer
3. Right Outer
4. Full Outer.

Refer [Supported join types](#) section to learn about the purpose of each join type.

![Join types](/static/assets/on-premise/images/report-designer/transforming-data/join-table/join-types.png)

In the below snap, **inner join** is created between the **PurchaseOrderDetail** and **SalesOrderDetail** tables.

![Choose inner join type](/static/assets/on-premise/images/report-designer/transforming-data/join-table/create-inner-join-between-tables.png)

Table fields

- **Left field** - The available columns of the table selected in the **Left Table** drop-down list will be listed here.

![Left field columns](/static/assets/on-premise/images/report-designer/transforming-data/join-table/left-field-columns-list.png)

- **Right field** - The available columns of the table selected in the **Right Table** drop-down list will be listed here.

![Right field columns](/static/assets/on-premise/images/report-designer/transforming-data/join-table/right-field-column-list.png)

In the below snap, the join field is created for **ProductID** column in both tables.

![Choose field name](/static/assets/on-premise/images/report-designer/transforming-data/join-table/join-fields-of-the-table.png)

Operator

To compare the values of the two columns (one from each table) between tables, any of the operator list shown in the below image can be used.

![Supported operators](/static/assets/on-premise/images/report-designer/transforming-data/join-table/list-of-supported-operators.png)

In the below snap, the **Equal** operator is applied between **ProductID** column of both tables.

![Choose operator](/static/assets/on-premise/images/report-designer/transforming-data/join-table/equal-operator.png)

Create multiple join condition

If you want to create multiple join condition for single table relation follow the below steps:

1. Click on the **Add Field** button to create multiple join conditions for single table relation.

![Add field button](/static/assets/on-premise/images/report-designer/transforming-data/join-table/add-field-button.png)

2. Choose the new column field in **Left Field** and **Right Field** drop-down list, then choose the operator condition.

![Create new join condition](/static/assets/on-premise/images/report-designer/transforming-data/join-table/new-join-condition.png)

3. Now, a new join condition will be created like below.

![Multiple join condition](/static/assets/on-premise/images/report-designer/transforming-data/join-table/multiple-join-condition.png)

Save a table relation

1. Click on the tick icon in the first row of table relation.

![Save icon](/static/assets/on-premise/images/report-designer/transforming-data/join-table/save-table-relation.png)

2. Click on the **OK** button.

![Save table relation](/static/assets/on-premise/images/report-designer/transforming-data/join-table/save-table-relation-output.png)

3. Now, the second table will be dropped in the design area like below.

![Query designer view](/static/assets/on-premise/images/report-designer/transforming-data/join-table/created-relationship-between-two-tables.png)

Edit a table relation

1. Click on the **Join** icon in the toolbar to open the **Query Joiner** dialog.

![Query joiner dialog](/static/assets/on-premise/images/report-designer/transforming-data/join-table/save-table-relation-output.png)

2. To edit the existing join condition in a table relation, click on the **Edit** icon in the respective field.

![Edit join relation icon](/static/assets/on-premise/images/report-designer/transforming-data/join-table/edit-join-icon.png)

3. Clicking on the icon will enable the respective join fields.

![Edit the join conditions](/static/assets/on-premise/images/report-designer/transforming-data/join-table/multiple-join-condition.png)

4. Edit the join condition as required and click on the **tick** icon to update the edited join condition.
5. Then, click on the **Ok** button to save the modified join relationship.

Delete a join condition

To delete a join condition in a table relation follow the below steps.

1. Open the **Query Joiner** dialog.
2. Mouse hover on the respective join field, to enable the delete option.

![Delete icon](/static/assets/on-premise/images/report-designer/transforming-data/join-table/delete-icon-for-join-condition.png)

3. Click on the above highlighted icon, to remove the join condition.

![Delete a join condition](/static/assets/on-premise/images/report-designer/transforming-data/join-table/equal-operator.png)

4. Click on the **tick** icon to update the join condition and then click on the **Ok** button to save the table relation.

A table relation must have atleast one join condition.

Delete a table relation

1. Open the **Query Joiner** dialog.
2. Delete a table relation by clicking the close icon placed at right corners in first row of table relation .

![Delete table relation](/static/assets/on-premise/images/report-designer/transforming-data/join-table/delete-icon-for-table-relation.png)

3. Click on the **Ok** button to save the joiner state.

Create multiple table relation

1. Drag and drop minimum three tables into the design area
2. Open the **Query Joiner** dialog and click on the **ADD** icon to create a new table relation.

![Add new relation](/static/assets/on-premise/images/report-designer/transforming-data/join-table/add-icon-new-table-relation.png)

3. Choose the new tables and join information in the fields and click on the tick icon.

![Choose fields to create new relation](/static/assets/on-premise/images/report-designer/transforming-data/join-table/multiple-table-relation.png)

4. Click on the **OK** button to save the joiner state.

Supported Join Types

The following sections describes the supported join types list and purpose of the each join type.

Inner join

INNER JOIN will return the records from two or more tables, while records are matching in both the tables.

An inner join of Table1 and Table2 gives the result of Table1 intersect Table2.

Supported Join Types

Left outer join

For example, consider the below two tables.

Table1

| SupplierId | SupplierName |
|------------|--------------|
| 100 | James |
| 101 | John |
| 102 | Robert |
| 103 | Michael |

Table2

| OrderId | SupplierId | Order_Date |
|---------|------------|------------|
| 20125 | 100 | 09/21/2017 |
| 20126 | 101 | 09/22/2017 |
| 20127 | 104 | 09/23/2017 |

If we join (INNER JOIN) Table1 and Table2 based on Supplier_Id column and equals (=) as comparison operator, then the result will be like below.

| SupplierId | SupplierName | OrderId | SupplierId(Table2) | Order_Date |
|------------|--------------|---------|--------------------|------------|
| 100 | James | 20125 | 100 | 09/21/2017 |
| 101 | John | 20126 | 101 | 09/22/2017 |

Left outer join

LEFT OUTER JOIN will return all record from the left table and the matched records from the right table. The result is NULL from the right table, if there is no match.

For example, consider the below two tables.

Table1

| SupplierId | SupplierName |
|------------|--------------|
| 100 | James |
| 101 | John |
| 102 | Robert |
| 103 | Michael |

Table2

| OrderId | SupplierId | Order_Date |
|---------|------------|------------|
| | | |

Supported Join Types

Right outer join

| |
|--------------------------|
| 20125 100 09/21/2017 |
| 20126 101 09/22/2017 |
| 20127 104 09/23/2017 |

If we join (LEFT OUTER JOIN) Table1 and Table2 based on Supplier_Id column and equals (=) as comparison operator, then the result will be like below.

| SupplierId / SupplierName | OrderId | SupplierId(Table2) | Order_Date |
|----------------------------------------|---------|--------------------|------------|
| ----- ----- ----- ----- | | | |
| 100 James 20125 100 09/21/2017 | | | |
| 101 John 20126 101 09/22/2017 | | | |
| 102 Robert | | | |
| 103 Michael | | | |

Right outer join

Right outer join preserves the unmatched rows from the second (right) table, joining them with a NULL in the shape of the first (left) table.

For example, consider the below two tables.

Table1

| OrderID | CustomerID | EmployeeID | OrderDate | ShipperID |
|---------------------------------|------------|------------|-----------|-----------|
| ----- ----- ----- ----- ----- | | | | |
| 10308 2 7 1996-09-18 3 | | | | |
| 10309 37 3 1996-09-19 1 | | | | |
| 10310 77 8 1996-09-20 2 | | | | |

Table2

| EmployeeID | LastName | FirstName | BirthDate | Photo |
|----------------------------------------|----------|-----------|-----------|-------|
| ----- ----- ----- ----- ----- | | | | |
| 1 Davolio Nancy 12/8/1968 EmpID1.png | | | | |
| 2 Fuller Andrew 2/19/1952 EmpID2.png | | | | |
| 3 Leverling Janet 8/30/1963 EmpID3.png | | | | |

If we join (RIGHT OUTER) Table1 and Table2 based on EmployeeID column and equals (=) as comparison operator, then the result will be like below.

| EmployeeID | LastName | FirstName | OrderID | EmployeeID | OrderDate | ShipperID |
|--------------------------------------------|----------|-----------|---------|------------|-----------|-----------|
| ----- ----- ----- ----- ----- ----- ----- | | | | | | |
| 1 Davolio Nancy 12/8/1968 | | | | | | |
| 2 Fuller Andrew 2/19/1952 | | | | | | |
| 3 Leverling Janet 8/30/1963 3 1996-09-19 2 | | | | | | |

Full outer join

The FULL OUTER JOIN keyword return all records when there is a match in either left (table1) or right (table2) table records.

For example, consider the below two tables.

Table1

| CustomerID | Name | ContactName | City | PostalCode | Country |
|------------|---------|----------------|-------------|------------|---------|
| 1 | Alfreds | Maria | Berlin | 12209 | Germany |
| 2 | Ana | Ana Trujillo | México D.F. | 05021 | Mexico |
| 3 | Antonio | Antonio Moreno | México D.F. | 05023 | Mexico |

Table2

| OrderID | CustomerID | EmployeeID | OrderDate | ShipperID |
|---------|------------|------------|------------|-----------|
| 10308 | 2 | 7 | 1996-09-18 | 3 |
| 10309 | 37 | 3 | 1996-09-19 | 1 |
| 10310 | 77 | 8 | 1996-09-20 | 2 |

The FULL OUTER JOIN keyword returns all the rows from the left table (table1), and all the rows from the right table (table2).

| CustomerID | Name | ContactName | City | PostalCode | Country | OrderID |
|------------|---------|----------------|-------------|------------|---------|---------|
| 1 | Alfreds | Maria | Berlin | 12209 | Germany | |
| 2 | Ana | Ana Trujillo | México D.F. | 05021 | Mexico | 10308 |
| 3 | Antonio | Antonio Moreno | México D.F. | 05023 | Mexico | 10365 |
| | | | 10382 | | | |
| | | | 10351 | | | |

Formatting Columns

When designing the data in query designer, the columns in the data table can be organized based on your dataset design requirements.

Refer [Create Data](#) section for better understanding with the following sections. The below image showcases the initial view of query designer.

![Query Designer initial state](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/query-designer-initial-view.png)

Rename a column

The column names in data table can be renamed in query designer after retrieving the data from data source.

The column rename action will not affect the actual data in the data source.

Follow the below steps to rename a column in data table.

1. Drag a table or view from the left pane and drop it into the center pane labeled with **Drag and Drop tables here** like below.

![Drag and drop table into design area](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/drag-and-drop-a-table.png)

2. Execute the query and the data will display in preview grid as follows.

![Data preview before rename action](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/preview-grid-data-before-rename-action.png)

3. Double-click on the column name in the table to rename the respective column.

![Enable the column edit mode](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/double-click-on-column-name.png)

4. The edit mode will be enabled like below.

![Column name edit mode](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/column-name-edit-mode.png)

5. Now, modify the column name and press enter key to commit the modification done.

![Modify the column name](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/after-renaming-action.png)

6. Execute the query, the renamed column will display in preview grid as follows.

![Data preview after rename action](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/preview-grid-data-after-rename-action.png)

Exclude columns

To remove or avoid the unwanted columns from the data design after retrieving the data from data source, the **Exclude** option can be used.

Excluding the columns from data table will not affect the actual data in the data source.

Follow the below steps to exclude and include a column in data table.

1. Exclude the column that you requisite, by clicking the icon at left corner of the respective column name.

![Exclude icon](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/exclude-icon.png)

2. The excluded column will be represented in table design like below.

![Exclude action representation](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/exclude-action-denotation.png)

3. Now, execute the query and notice that **SalesOrderID** column will be removed in the data preview.

![Data preview after column exclude action](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/preview-data-after-exclude-action.png)

You can also exclude all the columns in a single click, by clicking on the icon at the table header.

![Exclude all columns](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/exclude-all-columns.png)

Clicking on the same icon again will include all the columns.

![Include all columns](/static/assets/on-premise/images/report-designer/transforming-data/formatting-columns/select-all-columns.png)

Minimum of one column must be in enabled state to create dataset.

Query Filters

Query Filters are used to filter out specific data in a database. The data can be filtered by adding and deleting a filters.

![open-filter-dialog](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/query-filters-dialog.png)

Add filters

1. To add a filter, Click on the **Add** icon.

![add-field-filters](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/add-filter.png)

2. Dataset fields are listed in the first drop-down list, choose the necessary field from the drop-down list.

![add-expression](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/choose-fields.png)

3. **Operator** types are listed in the second drop-down list.

![operators-in- filters](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/operators.png)

4. In the **Value**, enter the value to be filtered in the dataset fields.

![pass-value-in-filters](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/pass-value.png)

5. Click on **Include as parameter** checkbox, will include the query parameters.

![include-parameter](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/include-parameter.png)

6. To add multiple filters, follow steps 1 - 5.

![add-with-multiple-filter](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/multiple-filters.png)

7. Click **OK** and select **Execute** icon as shown below to view the filtered data in the data preview.

![datapreview](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/filtered-data.png)

8. When save the dataset, the query parameter will automatically included in report parameter as shown below.

![report-parameter](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/query-report-parameter.png)

Remove Filters

Click **Delete** icon in the right corner to remove the respective filters.

![delete-filter](/static/assets/on-premise/images/report-designer/transforming-data/query-filter/delete-a-filter.png)

Create or link dataset parameters

A dataset parameters are created based on the query parameters provided in the dataset query and also, an equivalent report parameters are created for each query variable. The linking between a dataset parameter and a report parameter will take place automatically. But, if a report parameter is renamed then we need to perform the link action manually.

Refer [Create Data](#) section for better understanding with the following sections. The below image showcases the initial view of query designer.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/query-designer-initial-view.png)

Open parameters dialog

Drag and drop a table into design area. Now, click on the **Parameter** icon in the query designer toolbar.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/query-parameter-icon.png)

Now, the **Parameters** dialog will be launched.

Create or link dataset parameters

Create a new dataset parameter

![Parameter dialog](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/query-parameter-dialog.png)

Create a new dataset parameter

1. To add a new dataset parameter, click on the **Add** icon.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/add-parameter.png)

2. **Parameter Name** : In the **Parameter Name** text box, provide the name for the parameter.
3. **Value** : In **Value** field, provide the value in the `=Parameters!SalesOrderID.Value` format.
4. Click **OK** to save the dataset parameter.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/provide-values-in-fields.png)

Repeat the above steps to create multiple dataset parameters.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/multiple-parameters.png)

Once you save the dataset, an equivalent report parameters will be created under the **PARAMETERS** panel like below.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/report-parameters-list.png)

The report parameter name is created using the value provided in **Value** field. If the value field is left empty when creating the dataset parameter then the report parameter name is created using the value provided in **Parameter Name** field.

Set Expression

The **Value** field in dataset parameter can contain an expression that evaluates to a value, to pass to the query parameter.

1. To set expression, click on the square icon and select **Expression**.

![Expression icon](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/expression-icon-value-field.png)

2. It will launch the expression dialog like below.

![Expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/expression-dialog.png)

3. Refer [Expression](#) section to know more about the handling expressions in Report Designer. Here, the `SalesOrderID` field is assigned as expression for `SalesOrderID` parameter.

![Assign data field as expression](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/aassign-data-fields.png)

4. The icon will be indicated in **Black color**, if the expression is applied to the value field.

![Expression indication](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/expression-set-indication.png)

Reset Expression

1. To reset the applied expression in **Value** field, click on the square icon and select **Reset**.

![Select reset expression](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/reset-expression-option.png)

2. The icon will be indicated in **White color**, after reset action.

![After reset expression](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/expression-reset-indication.png)

Reorder parameters

When executing a query, the result will be retrieved from database based on the order of the parameter in the query.

The below snap depicts the parameter list before reordering the parameters.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/before-reordering-parameters.png)

To change the order of the dataset parameters after creating it, follow the below steps.

1. Click and hold the icon in the left corner of **Parameter Name** field.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/gripper-icon-to-perform-drag-action.png)

2. Then drag and move the dataset parameter field to higher or lower position in the list.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/drag-start-action.png)

3. Drop the dataset parameter field in desired position.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/drag-action-demo.png)

The below snap depicts the parameter list after reordering the parameters.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/after-reorder-action.png)

Define query parameters

Remove parameter

Remove parameter

Click on the **Delete** icon in the right corner to remove the dataset parameter from the list.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/delete-query-parameter.png)

Link parameters manually

The following procedure is applicable for below scenario's:

- To link a query variable with an existing report parameter.
- To relink a report parameter with query parameter, if the report parameter is renamed.

Create query and report parameters

Refer [Create report parameter](#) section and create the required report parameters in your report.

Refer [Create query parameter](#) section and create the required query parameters.

Steps to link parameters

The following steps guides to link a query parameter with a report parameter.

1. Open the dataset **Parameters** dialog, the available query parameters in the dataset query will be automatically listed in the dialog.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/parameter-list-on-open-action.png)

2. In the list find the parameter name to which you want to link a report parameter.
3. **Value** Field: The available report parameters in the report will be listed in the **Value** field drop-down. Choose a report parameter to which you want to link the respective query parameter.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/value-field-drop-down.png)

4. Click on the **OK** button to save the parameter linking state.

In the below snap, ProductID report parameter is linked with a ProductID query parameter.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/create-or-link-dataset-parameters/link-paramter-name.png)

Define query parameters

In query designer, the query parameter can be created in both design and query mode. The following sections describes the steps to create query parameters in design and query mode.

Refer [Create Data](#) section for better understanding with the following sections. The below image showcases the initial view of query designer.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/query-designer-initial-view.png)

Create query parameter in design mode

1. Drag and drop a table into design area and open the **Query Filters** dialog, in the query designer.

![Open query filter dialog](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/open-query-filter-dialog.png)

Refer [Query filters](#) section to learn more about query filters.

2. Click on the **Add** icon, to create a new filter.
3. Choose any dataset field in the first drop-down list.
4. Choose any **Operator** in the second drop-down list.
5. In the **Value** field, provide a value to filter the data.
6. Select the checkbox next to the value filed, to create a new **query parameter**.
7. Click on the **OK** button.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/create-query-parameter.png)

Once you save the dataset, an equivalent report parameters will be created under the **PARAMETERS** panel like below.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/report-parameters-list.png)

Create query parameter in code mode

1. Click on the **switcher** in query designer toolbar, to switch to the query mode.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/code-switcher-icon.png)

2. Specify the query in the text area, the query specified in the below snap returns the list of data from the **AdventureWorks** database, **SalesOrderDetail** table for the **Sales** schema.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/specify-query-in-text-area.png)

3. Now, add the following **WHERE** clause at the end of the query to create a query parameter.

`js

WHERE (([Sales].[SalesOrderDetail].[SalesOrderID] = @SalesOrderID))

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/add-where-clause.png)

4. Click on the **Finish** button, now the **Parameters** dialog opens.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/query-parameter-dialog.png)

5. Enter the value for parameter in the value field and click **OK**.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/enter-query-parameter-value.png)

Once you save the dataset, an equivalent report parameters will be created under the **PARAMETERS** panel like below.

![Query joiner icon in enabled state](/static/assets/on-premise/images/report-designer/transforming-data/define-query-parameters/report-parameters-list.png)

Query expression

An expression column is used to create an expression which is a combination of data columns, operators and built-in functions. This expression column will act as a calculated measure that can be configured to report item like other normal numeric columns as a quantitative measure.

Open query expression dialog

Click on the **Expression** icon in the query designer toolbar to open **Query Expressions** dialog.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/query-expression-icon.png)

Now the **Query Expressions** dialog will be launched like below.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/query-expression-dialog.png)

Note: The expression icon in the tools pane will be in disabled state, if there is no table found dropped in table design view.

Create an expression column

1. To create a new expression column, click on the **Add** button in the **Query Expressions** dialog .
2. In **Name** field, type the name for the expression column or use the default name. By default, manually-created expression columns name are similar to **Expression1**.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/expression-name-field.png)

3. Define an expression in an **Expression** text area. The syntax for defining a simple expression is,

{function name() [columnname] {operator [columnname]} ...}

where, content within curly braces is optional.

Some expressions for reference:

1. **YEAR([Order Date])** – To compute year of order date.

2. $[Freight]+100$ – To compute the total with 100 added to Freight.
3. To include the function names and the column names in the Expression text area, double click on the respective function or column name in the Functions or Column Settings list view.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/list-view.png)

5. After, designing the expression value in text area, click on the Save button.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/save-expression.png)

6. To create multiple expression columns, repeat the above steps.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/create-multiple-expression.png)

7. The newly added expressions will be listed under the Column Settings list view like below.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/expressions-listed-in-list-view.png)

Refer [Supported Built-in functions](#) section to know more about supported built-in functions.

Edit an expression column

1. Select an expression column in left pane, which you want to update.
2. Edit the Name and Expression fields as required.
3. Click on the Save button to save the changes.

Delete an expression column

Click on the delete icon in the expression field to delete an expression column.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/delete-expression.png)

Close query expression dialog

To close the dialog click on the Cancel button in dialog footer or Close icon in the dialog header.

![Query expression dialog](/static/assets/on-premise/images/report-designer/transforming-data/configuring-expression-columns/close-query-expression-dialog.png)

Supported built-in functions

Following built-in functions are supported in Query Expression Designer.

Numbers

| Functions | Syntax & Descriptions |
|-----------|-----------------------|
| | |

| | |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ABS | Syntax: ABS(numeric_expression) Description: Returns the absolute value of the given expression. |
| ACOS | Syntax: ACOS(numeric_expression) Description: Returns the inverse cosine (also known as arccosine) of the given numeric expression. |
| ASIN | Syntax: ASIN(numeric_expression) Description: Returns the inverse sine (also known as arcsine) of the given numeric expression. |
| ATAN | Syntax: ATAN(numeric_expression) Description: Returns the inverse tangent (also known as arctangent) of the given numeric expression. |
| COS | Syntax: COS(numeric_expression) Description: Returns the cosine of the angle specified in radians, in the given expression. |
| DEGREES | Syntax: DEGREES(numeric_expression) Description: Returns the angle in degrees for the one specified in radians, in the given numeric expression. |
| EXP | Syntax: EXP(numeric_expression) Description: Returns the exponential value of the given expression. |
| LOG | Syntax: LOG(numeric_expression) Description: Returns the logarithm of the given expression to the specified base. |
| PI | Syntax: PI() Description: Returns the constant value of PI. |
| POWER | Syntax: POWER(numericexpression, numericexpression) Description: Returns the value of the given expression (expression1) to the specified power (expression2). |
| ROUND | Syntax: ROUND(numeric_expression) Description: Returns a rounded value. |
| RADIANS | Syntax: RADIANS(numeric_expression) Description: Returns the angle in radians for the one specified in degrees in the given numeric expression. |
| SIGN | Syntax: SIGN(numeric_expression) Description: Returns a value representing the positive (+1), zero (0), or negative (-1) sign of the given numeric expression. |
| SIN | Syntax: SIN(numeric_expression) Description: Returns the sine of the angle specified in radians, in the given expression. |
| SQRT | Syntax: SQRT(numeric_expression) Description: Returns the square root of the given numeric expression. |
| TAN | Syntax: TAN(numeric_expression) Description: Returns the tangent of the given numeric expression. |

Conditional

| Functions | Syntax & Descriptions |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| IF | Syntax: IF(expression, truepart, falsepart) Description: Returns either true part or false part, depending on the evaluation of the expression. |

| | |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IFNULL | Syntax: IFNULL(expression1,expression2) Description: If the expression is numeric/string/date, returns the first expression. If the first expression is NULL, returns the second expression. |
| ISNOTNULL | Syntax: ISNOTNULL(expression) Description: If the expression is numeric/string/date is NULL, returns a string representing false, otherwise returns true. |
| ISNULL | Syntax: ISNULL(expression) Description: Returns true if the given expression evaluates to null. |

Logical

| Functions | Syntax & Descriptions |
|-----------|---------------------------------------------------------------------------------------------------------------|
| AND | Syntax: (expression1) AND (expression2) Description: Returns true if both the expressions evaluates to true. |
| NOT | Syntax: NOT(expression) Description: Returns the reversed logical value of the expression being evaluated. |
| OR | Syntax: (expression1) OR (expression2) Description: Returns true if any of the expressions evaluates to true. |

Date

| Functions | Syntax & Descriptions |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| DATEADD | Syntax: DATEADD(numericexpression, dateexpression) Description: Adds the number of days to the specified date. |
| DATENAME | Syntax: DATENAME(datepart, dateexpression) Description: Returns a string representing the specified date_part of the given date expression. |
| DATEPART | Syntax: DATEPART(datepart, dateexpression) Description: Returns an integer value representing the specified date_part of the given date expression. |
| DATESUB | Syntax: DATESUB(numericexpression, dateexpression) Description: Returns the date subtracted from the specified date. |
| DAY | Syntax: DAY(date_expression) Description: Returns a numeric value representing the day part of the specified date. |
| DAYDIFF | Syntax: DAYDIFF(startdateexpression, enddateexpression) Description: Returns a numeric value representing the difference between two specified dates. |
| HOUR | Syntax: HOUR(date_expression) Description: Returns the hour of the given date as an integer. |
| MINUTE | Syntax: MINUTE(date_expression) Description: Returns a numeric value representing the minute part of the date resulted from specified date expression. |

| | |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MONTH | Syntax: MONTH(date_expression) Description: Returns a numeric value representing the month part of the date resulted from specified date expression. |
| NOW | Syntax: NOW() Description: Returns the current date and time. |
| TODAY | Syntax: TODAY() Description: Returns the current date. |
| YEAR | Syntax: YEAR(date_expression) Description: Returns a numeric value representing the year part of the date resulting from the specified date expression. |
| MAX | Syntax: MAX(expression) Description: Returns the maximum value in the given expression. |
| MIN | Syntax: MIN(expression) Description: Returns the minimum value in the given expression. |

String

| Functions | Syntax & Descriptions |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CHAR | Syntax: CHAR(integer_expression) Description: Converts the given integer ASCII code into a character. |
| CONCAT | Syntax: CONCAT(stringexpression1, stringexpression2, ..., string_expressionN) Description: Returns a string value resulting from the concatenation of two or more string values. |
| CONTAINS | Syntax: CONTAINS(stringexpression, substringexpression) Description: Returns true if the given string expression contains the specified substring expression. |
| ENDSWITH | Syntax: ENDSWITH(stringexpression substringexpression) Description: Returns true if the given string expression ends with the specified substring expression. |
| LEFT | Syntax: LEFT(stringexpression, numericexpression) Description: Returns the specified number of characters from start of the given string expression. |
| LEN | Syntax: LEN(string_expression) Description: Returns the number of characters in the given string expression. |
| LOWER | Syntax: LOWER(string_expression) Description: Returns a lower case converted string value from a given string expression. |
| LTRIM | Syntax: LTRIM(string_expression) Description: Returns the string value with any leading blanks removed from string expression. |
| MAX | Syntax: MAX(expression) Description: Returns the maximum value in the given expression. |
| MIN | Syntax: MIN(expression) Description: Returns the minimum value in the given expression. |
| RIGHT | Syntax: RIGHT(stringexpression, numericexpression) Description: Returns the specified number of characters from end of the given string expression. |

| | |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RTRIM | Syntax: RTRIM(string_expression) Description: Returns the string value with any trailing blanks removed from string expression. |
| STARTSWITH | Syntax: STARTSWITH(stringexpression, substringexpression) Description: Returns true if the given string expression starts with the specified substring expression. |
| SUBSTR | Syntax: SUBSTR(stringexpression, startingindex, lengthofthe_string) Description: Returns a specific length of string starting from specific index from the given string expression. |
| UPPER | Syntax: UPPER(string_expression) Description: Returns an upper case converted string value from a given string expression. |

Report Parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it. Parameters can also be used in expressions to control report data.

Data set query parameters

Allows creating an embedded or a shared dataset with parameter to filter or limit data. A data set can contain user-defined parameters for internal calculations or report parameter to get run-time inputs from users.

Cascading parameter

Users can add cascading parameters to the report to allow one parameter that limits the values for the next parameter. It greatly reduces the number of choices to a manageable number.

See also

[Add report parameter](#)

[Edit report parameter](#)

[Delete report parameter](#)

[Cascading parameter](#)

[Multi-value parameter](#)

Add report parameter to the report

Report parameters can be used to filter the report data or associate related reports together. This section guides you to add a report parameter into the report with Bold Report Designer.

Create parameter

- To add a parameter, click on the **Parameter** icon in the **Data Configuration** panel. It opens the **PARAMETERS** panel.

![Parameter icon](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/parameter-icon.png)

2. Next, click on the **NEW PARAMETER** button in the **PARAMETERS** panel.

![Parameters pane](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/parameters-pane.png)

3. Now, the following wizard will be displayed.

![Parameter configuration panel](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/parameter-configuration-panel.png)

- In **Name** field, type the name for the parameter or use the default name. By default, manually-created parameters name are similar to **ReportParameter1**.
- In **Prompt** field, type the text that appears next to the parameter when the user previews the report.
- In **Data type** field, select the data type for the parameter value. The supported data types are **Boolean**, **DateTime**, **Integer**, **Float** and **Text**.
- Select **Allow blank value**, if parameter value needs to be set as an empty string or a blank value.

Note: If you specify valid values for a parameter, and you want a blank value to be one of the valid values, you must include it as one of the values that you specify. Selecting this option does not automatically include a blank value for available values.

- Select **Allow null value**, if the value of the parameter needs to be set as null.

Note: If you specify valid values for a parameter, and you want null to be one of the valid values, you must include null as one of the values that you specify. Selecting this option does not automatically include the null value for available values.

- Select **Allow multiple values**, if the value for the parameter should be multiple values. Null values are not allowed.
- Set the **visibility** option.
 1. Select **Visible** to display the report parameter at the top of the report while running the report. This option allows you to select parameter values at run time.
 2. Select **Hidden** to hide the report parameter in the published report. The report parameter values can still be set on a report URL, in a subscription definition, or on the Report Server.
 3. Select **Internal** to hide the report parameter. In the published report, the report parameter can only be viewed in the report definition.

![Create new parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-new-parameter.png)

4. Click on **Save** button. Now, the parameter will be listed under the **PARAMETERS** pane like below.

![Product id parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/product-id-parameter.png)

Filter a table data based on report parameter

Using the **ProductID** parameter, we can filter the **ProductID** field values at runtime like below.

1. Select the table report item to enable grouping panel in the designer.

![Open table data assign menu](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/enable-grouping-panel.png)

2. Now, click on the **(Details)** member field in the grouping panel to open the tablix member properties.

![Open tablix member properties](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/tablix-member-properties.png)

3. In **Tablix Member** properties, click on **Set Filters...** button.

![Open filter dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/tablix-member-set-filters.png)

4. Refer [Set filters](#) section to create new filter expression.

![Create filter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-product-id-filter.png)

5. Choose the **Equal** operator and assign the **ProductID** parameter in the **Value** field. Save the filter.

![Save filter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/save-filter-equation.png)

6. Preview the report, now the following view will be displayed.

![Textbox parameter type](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/textbox-parameter-type.png)

7. Enter the valid **ProductID** in the textbox and click on **View Report**.

![Filter product id values](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/filter-product-id-values.png)

Edit a report parameter

This section describes the steps required to edit a report parameter in Bold Report Designer.

1. In the configuration panel, click on the **Parameters** icon to open the **PARAMETERS** panel.
2. Hover the cursor on the parameter which you want to edit from the list of parameters.

Delete a report parameter

Filter a table data based on report parameter

![Parameter list view](/static/assets/on-premise/images/report-designer/report-parameters/edit-report-parameter/parameter-list-view.png)

3. Click the **menu** icon in the right corner, to open the context menu.

![Open context menu](/static/assets/on-premise/images/report-designer/report-parameters/edit-report-parameter/open-context-menu.png)

4. Choose **Edit** from the context menu, it will launch the **EDIT PARAMETER** panel.

![Edit report parameter](/static/assets/on-premise/images/report-designer/report-parameters/edit-report-parameter/edit-report-parameter.png)

5. Modify the parameter fields and click **Save**.

Delete a report parameter

This section describes the steps required to remove a report parameter in Bold Report Designer.

1. In the configuration panel, click on the **Parameters** icon to open the **PARAMETERS** panel.
2. Hover the cursor on the report parameter which you want to delete from the list of parameters.

![Parameter list view](/static/assets/on-premise/images/report-designer/report-parameters/delete-report-parameter/parameter-list-view.png)

3. Click the **menu** icon in the right corner, to open the context menu.

![Open context menu](/static/assets/on-premise/images/report-designer/report-parameters/delete-report-parameter/open-context-menu.png)

4. Choose **Delete** from the context menu, it will launch the confirmation dialog like below.

![Parameter delete confirmation dialog](/static/assets/on-premise/images/report-designer/report-parameters/delete-report-parameter/parameter-delete-confirmation-dialog.png)

5. Click on **Yes** button, now the report parameter will be removed from the report parameter list.

Define available values for a parameter

An available values can be specified for a report parameter to allow the user to select only valid values on report preview action. The available values defined for the parameter will be listed in the drop-down list when previewing the report.

Refer [Create Parameter](#) section for better understanding with the following steps.

![Create new parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-new-parameter.png)

Click on **Assign Value >>** to open parameter assign dialog.

![Parameter assign values dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/parameter-assign-values-dialog.png)

Manual values

Follow the below steps to add available values for the parameter.

1. Select **Specify** option under Available Value tab.

![Parameter assign values dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/available-value-specify-value.png)

2. Click on the **Add** icon. Now, a list in which you can type values and labels appears.

![Available value add field](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/available-value-add-field.png)

3. Enter the value in the **Value** text box, and optionally, the label in the **Label** text box.

If you do not provide the label, the value is used.

![Create available values list](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-available-values-list.png)

4. Click **OK** and **Save** the parameter.

![Save report parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/save-report-parameter.png)

On report preview, the available values defined for the parameter will be listed in the drop-down list like below.

![Available values list](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/drop-down-values-list-specify-option.png)

Query values

1. Select **Query Value** option under Available Value tab.

![Available value query value option](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/available-value-query-value-option.png)

2. In **Dataset** drop-down, choose the name of the dataset. Datasets can be defined using the data view.
3. In **Value field**, choose the name of the field that provides parameter values.

Note: These fields are retrieved from the list of column or field names in the dataset.

4. In **Label field**, choose the name of the field that provides the parameter names. If there is no separate field for names, choose the same field similar to **Value field**.

Define available values for a parameter

Remove available values

5. Click **OK**.

![Available values query value fields](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/available-values-query-value-fields.png)

6. Save the parameter.

On report preview, the **ProductID** parameter will list the values of **ProductID** field from query data.

![Query value drop down output](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/query-value-drop-down-output.png)

Remove available values

To remove the available values defined for the parameter follow the below steps:

1. Refer [Edit Parameter](#) section and open parameter properties.
2. Click on **Assign Value >>** to open parameter assign dialog.
3. Under the **Available Values** tab, choose the **None** option.

![Remove available values](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/remove-available-values.png)

4. Click **OK** and save the parameter.

Filter a table data based on report parameter

Using the **ProductID** parameter, we can filter the **ProductID** field values at runtime like below.

1. Select the table report item to enable grouping panel in the designer.

![Open table data assign menu](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/enable-grouping-panel.png)

2. Now, click on the **(Details)** member field in the grouping panel to open the tablix member properties.

![Open tablix member properties](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/tablix-member-properties.png)

3. In **Tablix Member** properties, click on **Set Filters...** button.

![Open filter dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/tablix-member-set-filters.png)

4. Refer [Set filters](#) section to create new filter expression.

![Create filter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-product-id-filter.png)

5. Choose the **Equal** operator and assign the **ProductID** parameter in the **Value** field. Save the filter.

![Save filter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/save-filter-equation.png)

6. Preview the report, now the following view will be displayed.

![Textbox parameter type](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/drop-down-list-parameter-using-specify-option.png)

7. The manually created values will be listed in the drop-down. Choose any value from the drop-down list.

![Textbox parameter type](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/drop-down-values-list-specify-option.png)

8. Click on **View Report**.

![Filter product id values](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/preview-of-drop-down-list-parameters-using-specify-option.png)

Define default values for a parameter

A list of default values can be specified for a report parameter. If the parameters in the report have default value, the report runs automatically on report preview action.

Refer [Create Parameter](#) section for better understanding with the following steps.

![Create new parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-new-parameter.png)

Click on **Assign Value >>** to open parameter assign dialog.

![Parameter assign values dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/parameter-assign-values-dialog.png)

By default, the parameter dialog will be launched with **Available Value** tab. To switch over to **Default Value** tab, click the **Default Value** which has below options.

![Default value option](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-value-option.png)

Manual values

1. Select **Specify** option under **Default Value** tab.

![Default specify option](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-specify-option.png)

2. Click on the **Add** icon. Now, a list in which you can type values appears.

![Default tab add value](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-tab-add-value.png)

3. Enter the value in the **Value** text box and click **OK**

![Default tab enter valid value](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-tab-enter-valid-value.png)

4. Save the parameter.

On report preview, the report automatically runs and displays all the records whose **ProductID** is **712**.

![Default manual values output](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-manual-values-output.png)

Query values

1. Select **Query Value** option under **Default Value** tab.

![Default tab query value](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-tab-query-value.png)

2. In **Dataset** drop-down, choose the name of the dataset. Datasets can be defined using the data view.
3. In **Value field**, choose the name of the field that provides parameter values.

Note: These fields are retrieved from the list of column or field names in the dataset.

4. Click **OK**.

![Default value assign data](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-value-assign-data.png)

5. Save the parameter.

On report preview, the report automatically runs and displays the first record of **ProductID** data field.

![Default tab query value output](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/default-tab-query-value-output.png)

Remove default values

To remove the default values defined for the parameter follow the below steps:

1. Refer [Edit Parameter](#) section and open parameter properties.
2. Click on **Assign Value >>** to open parameter assign dialog.
3. Under the **Default Values** tab, choose the **None** option.

![Remove default values](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/remove-default-values.png)

4. Click **OK** and save the parameter.

Create a multi value report parameter

Multi Value Parameter is used to dynamically filter the report data based on more than one value. This section describes the steps required to create a multi value report parameter in Bold Report Designer.

Refer [Create Parameter](#) section for better understanding with the following steps.

![Create new parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/create-new-parameter.png)

1. Select **Allow multiple values**, to create a multi-value parameter.

A multi-value parameter cannot include **null** values.

![Create new parameter](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/enable-allow-multiple-values-option.png)

2. Click on **Assign Value >>** to open parameter assign dialog.

![Parameter assign values dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/parameter-assign-values-dialog.png)

3. Select **Query Value** option under **Available Value** tab.

![Available value query value option](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/available-value-query-value-option.png)

4. Select the dataset and fields in the drop-down list and Click **OK**.

![Available values query value fields](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/available-values-query-value-fields.png)

5. Save the parameter.

Filter a table data based on report parameter

Using the **ProductID** parameter, we can filter the **ProductID** field values at runtime like below.

1. Select the table report item in design area.

![Select the table report item](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/select-table-report-item.png)

2. Click on the **Properties** icon in the configuration panel, to open table properties.

![Open table properties](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/properties-icon-to-open-table-properties.png)

3. In **Table** properties, click on **Set Filters...** button.

![Table set filters button](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/table-set-filters-button.png)

4. Refer [Set filters](#) section to create new filter expression.
5. Choose the **In** operator and assign the **ProductID** parameter in the **Value** field. Save the filter.

![Open expression menu in filter dialog](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/add-filter-condition.png)

6. Click on the **Preview** button in the report header. Select the required values in drop-down list.

![Select multiple values to filter the records](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/select-multiple-value-in-parameter.png)

7. Click on the **View Report** button.

![Multi value parameter preview](/static/assets/on-premise/images/report-designer/report-parameters/add-report-parameter/multi-value-parameter-preview-output.png)

Cascading Parameter

The concept of cascading parameters is a list of values for one parameter depends on the values chosen in another parameter. This type of parameters can be used when a parameter has a long list of values.

Steps to create cascading parameters

The following section describes the step by step process to create a cascading parameters in Web Report Designer.

Create the main dataset

Create a new dataset by following the steps provided in [Create Data](#) section before proceeding with below steps:

1. In **Name** field, type the name of the dataset. For example, we have provided **SalesbyCategory** as dataset name.
2. Click on the **switcher** in query designer toolbar, to switch to the query mode.

![Code switcher icon](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/code-switcher-icon.png)

3. The query must have the following parts:
 - o A list of column fields in **SELECT** statement to fetch from any specific table.
 - o One query parameter for each cascading parameter.

For example, we have chosen tables **SalesPerson**, **SalesOrderHeader**, **SalesOrderDetail**, **Product**, **ProductSubcategory**, **ProductCategory** from AdventureWorks database to include query parameters @Category and @Subcategory in below query:

```
js
SELECT
PC.Name AS Category,
PSC.Name AS Subcategory,
SOH.[OrderDate],
SOH.SalesOrderNumber,
SD.OrderQty,
SD.LineTotal
FROM [Sales].[SalesPerson] SP
INNER JOIN [Sales].[SalesOrderHeader] SOH
ON SP.[SalesPersonID] = SOH.[SalesPersonID]
INNER JOIN Sales.SalesOrderDetail SD
ON SD.SalesOrderID = SOH.SalesOrderID
INNER JOIN Production.Product P
ON SD.ProductID = P.ProductID
INNER JOIN Production.ProductSubcategory PSC
ON P.ProductSubcategoryID = PSC.ProductSubcategoryID
INNER JOIN Production.ProductCategory PC
ON PC.ProductCategoryID = PSC.ProductCategoryID
WHERE (PC.Name = (@Category)
AND PSC.Name = (@Subcategory))
`
```

Paste the above query in query editor.

![Query of Sales by category dataset](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/sales-by-category-dataset-query.png)

- Click on the Run icon in toolbar, now the **Parameters** dialog opens automatically. Type the desired value for each query parameter in the parameter value column.

For example,

- In **@Category** parameter, we have typed Components as value.
- In **@Subcategory** parameter, we have typed Brakes as value.

![Query parameter dialog](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/sales-by-category-parameters-dialog.png)

5. Click on the **Finish** button. Now, the **SalesbyCategory** dataset will be listed under the **DATA** pane.

Once you save the dataset, an equivalent report parameters will be created under the **PARAMETERS** panel like below.

![Report parameter list for sales by category dataset](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/sales-by-category-parameter-list.png)

Create dataset for independent parameter

1. Create a new dataset by following the steps provided in [Create Data](#) section before proceeding with below steps.
2. In **Name** field, type the name of the dataset. For example, we have chosen **CategoryValues** as dataset name.
3. Paste the following query text in the query editor:

```
js
```

```
SELECT DISTINCT Name AS Category FROM Production.ProductCategory
```

```
\
```

Here column name **Name** and table **ProductCategory** has been represented in above query to create dataset for independent parameter **Category**.

![Category values dataset](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/category-values-dataset.png)

4. Click on the **Run** icon in toolbar to see the result.

![Category values dataset](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/category-values-dataset-preview-data.png)

5. Click on the **Finish** button. Now, the **CategoryValues** dataset will be listed under the **DATA** pane.

Set available values for independent parameter

1. Click **Parameter** icon in the configuration panel to open a **PARAMETERS** panel.

![Parameter icon](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/parameter-icon.png)

2. Edit the **Category** parameter. In name field, verify that the name is **Category**. Refer [Edit Report Parameter](#) section.
3. Click **Assign Value >>** to open the **Parameter Assign** dialog.

4. Select **Query Value** option under **Available Value** tab.
5. Select the **CategoryValues** in dataset field, **Category** in value and label field.

![Define available values for category parameter](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/category-parameter-query-value.png)

6. Click on the **Ok** button and **Save** the parameter.

Create dataset for dependent parameter

Next, need to create dataset for dependent parameter and assign values to it. For example, we have chosen **@Subcategory** as dependent parameter.

1. Create a new dataset by following the steps provided in [Create Data](#) section before proceeding with below steps.
2. In **Name** field, type the name of the dataset. For example, we have chosen **SubcategoryValues** as dataset name.
3. Switch to the query editor mode and paste the following query text in the query editor:

```
'js
SELECT DISTINCT PSC.Name AS Subcategory
FROM Production.ProductSubcategory AS PSC
INNER JOIN Production.ProductCategory AS PC
ON PC.ProductCategoryID = PSC.ProductCategoryID
WHERE PC.Name = (@Category)
'
```

Here column name **Name** and table **ProductSubCategory** joins with table **ProductCategory** and includes condition with independent parameter **Category** to create dataset for dependent parameter **SubCategory**.

![Sub category values dataset](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/sub-category-values-dataset-query.png)

4. Click on the **Run** icon in toolbar, now the **Parameters** dialog opens automatically. For example, In **@Category** parameter, we have typed **Components** as value.

![Query parameters dialog](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/query-parameter-dialog-for-category-parameter.png)

5. Click on the **OK** button, the result set will display 14 rows.

![Preview data for sub category values dataset](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/sub-category-values-preview-data.png)

6. Click on the **Finish** button. Now, the **SubcategoryValues** dataset will be listed under the **DATA** pane.

Set available values for dependent parameter

1. Click on the **Parameter** icon in the configuration panel to open a **Parameter** configuration panel.
2. Edit the **Subcategory** parameter. In name field, verify that the name is **Subcategory**.
3. Click on **Assign Value>>** to open the **Parameter Assign** dialog.
4. Select **Query Value** option under **Available Value** tab.
5. Select the **SubcategoryValues** in dataset field, **Subcategory** in value and label field.

![Define available values for Subcategory parameter](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/subcategory-parameter-query-value.png)

6. Click on the **Ok** button and **Save** the **Subcategory** parameter.

Filter a table data based on report parameter

1. Design a simple table report and assign the **SalesbyCategory** dataset to the table.

![Design table report](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/simple-table-report-design.png)

Refer [Design a simple table report](#) section to create a table report in Web Report Designer.

2. Click on the **Preview** button in the report header.
3. Choose **Accessories** in the **Category** parameter drop-down, based on the **Accessories** category the values will be populated in the **SubCategory** drop-down.

![Choose value for Category parameter](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/sub-category-parameter-drop-down.png)

4. Choose **Bike Racks** in the **Subcategory** dropdown and click on **View Report** button.

![Final report preview](/static/assets/on-premise/images/report-designer/report-parameters/create-cascading-parameter/report-preview-final-view.png)

Parameters Layout

Parameters layout is a grid layout where you can arrange the order of parameters. When you preview the report, the order of parameters displayed on the parameters pane is determined by the order of the parameters in the parameters layout and parameters pane.

Parameters order is important when designing a reports with cascading parameters.

You can access the parameters layout by using the **Edit Layout** option in **Parameters Pane**.

1. Open the parameters pane in the navigation panel.

![Open parameters pane](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/open-parameters-pane.png)

2. Click on **Edit Layout >** option in the parameters pane.

![Edit layout option to open layout](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/edit-layout.png)

3. Now, the parameters layout dialog will open.

![Parameters layout dialog](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/parameters-layout-dialog.png)

By default, the layout opens with four columns and two rows. You can manually reorder parameters, insert or delete rows and columns in the parameters layout dialog.

The parameter layout supported for reports created from RDL XML schema version 2016 onward.

Order parameters

In the grid, you can move parameters to any position by dragging them to the desired cell. If there is already a parameter in that cell, the parameters swap positions. Changing the position of a parameter also impact the order of the parameter list in parameters pane. Now, the parameters layout have four parameters in a row with four columns. In report preview, the parameter pane will look like below,

![Before ordering parameters](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/before-ordering-in-viewer.png)

To arrange the parameter positions drag a required parameter from a cell and drop into required cell position,

![Ordering parameters](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/swap-positions.png)

Then click OK. Notice the order of parameters in the parameters pane list also changed.

Now, I have arranged the parameters like below,

![After ordering parameters](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/after-arranging-positions.png)

Parameters list in the parameters pane also reordered based on the parameter layout,

![Ordering effect in parameters pane](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/reordering-in-list.png)

After ordering, at report preview the parameter pane will look like below,

![After reordering parameters in viewer](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/after-reordering-in-viewer.png)

Insert row or column

As in any grid-like structure, you can add new columns and rows or remove them. We can add maximum of 8 columns, but the number of rows appears not have a limit. To insert a row or column, right click on

a cell around which you want to add row or column. The context menu will show up with **Insert Column** and **Insert Row** options.

![Cell context menu](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/cell-context-menu.png)

You can add a column to the **Left** or **Right** of the target row.

![Insert row options](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/insert-column-options.png)

Similarly, you can add a row **Above** or **Below** the target row.

![Insert column options](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/insert-row-options.png)

Delete row or column

As like insert row and column actions, you can also delete a parameter or delete a row or column in the parameter layout. Deleting a row or column will also delete all associated parameters on that specific row or column. The delete action in parameters layout will also deletes the respective parameters from parameters pane. To delete a row or column, right click any cell in that respective row or column. The context menu will show up with **Delete Column** and **Delete Row** options.

![Cell context menu](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/cell-context-menu.png)

Performing delete action on an empty row or column will automatically deletes the respective row or column from the layout. If the row or column contains one or more parameters, it will show a alert confirmation like below.

![Alert confirmation](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/delete-alert-confirmation.png)

You can also delete a specific parameter from a row or column using the **Delete** option in the context menu. Right click on the cell in which you want to delete the parameter and select **Delete** option from the menu.

![Delete single parameter](/static/assets/on-premise/images/report-designer/report-parameters/parameters-layout/delete-single-parameter.png)

Once you click OK. It also deletes the respective parameter from parameters pane.

Image manager

Bold Report Designer supports to embed local or database images into the report and image data is stored within the report definition. When you embed an image, MIME-encodes the image and stores it as text in the report definition. The embedded images in a report are listed in the **Image Manager panel**.

See also

[Embed an image](#)

[Delete embedded image](#)

Image Manager

Image manager allows to add an embedded image to the report, the embedded image will always be available to that report. The embedded images in a report are listed in the **Image Manager** panel.

Add an embedded image

1. Click on the **Image Manager** icon in the configuration panel.

![hover-imagemanager](/static/assets/on-premise/images/report-designer/imagemanager/icon.png)

2. Click on **Add Image** button in the **IMAGE MANAGER** panel.

![imagemanager-panel](/static/assets/on-premise/images/report-designer/imagemanager/panel.png)

3. Select an image and click on **Open** button.

![upload-dialog](/static/assets/on-premise/images/report-designer/imagemanager/add-image-dialog.png)

4. Now, the image will be embedded to the report and listed under the **IMAGE MANAGER** panel.

![upload-single-image](/static/assets/on-premise/images/report-designer/imagemanager/upload-image-in-imagemanager.png)

Note: Only one image can be imported at a time. To add more images repeat steps 2 to 4.

![upload-multiple-image](/static/assets/on-premise/images/report-designer/imagemanager/images-in-list-view.png)

Design a report using an embedded image

1. To design a report using an embedded image, mouse hover on the image in the **IMAGE MANAGER** panel.

![hover-image](/static/assets/on-premise/images/report-designer/imagemanager/hover-an-image.png)

2. Click on the **plus** icon, now the image item will be added to the design area.

![add-to-designer-surface](/static/assets/on-premise/images/report-designer/imagemanager/add-image-to-designArea.png)

3. The below image shows the simple report designed using an embedded image.

![add-image-to-header-footer](/static/assets/on-premise/images/report-designer/imagemanager/image-in-header-footer.png)

Delete an embedded image

1. To delete an image listed under the IMAGE MANAGER pane, mouse hover on the image which you want to delete.

![delete-image-icon](/static/assets/on-premise/images/report-designer/imagemanager/delete-an-image.png)

2. Click on the above highlighted Delete icon, it will launch the confirmation dialog like below.

![delete-alert-dialog](/static/assets/on-premise/images/report-designer/imagemanager/delete-image-alert.png)

3. Click on the Yes button to remove an image from the report.

Compose report

Toolbar

The Report Designer toolbar contains a set of icons or buttons that allows you to perform common report designing operations.

Refer [Toolbar](#) section to handle the options in toolbar.

Properties panel

The properties pane significantly simplifies report setup and styling. Allows customizing the appearance of report items and editing their property values. Users can specify static values or expressions as values in the report item properties.

Refer [Properties panel](#) section to set and style report items.

Multi select report item properties

You can edit common properties of multiple report items at once. Tablix item have styling options, such as apply formatting by selecting specific cells, rows, or columns.

Refer [Report item multiselection](#) section to edit properties of multiple report items.

Filtering

Provides support to add filters to an embedded data set or shared data set, data regions, and data region groups, including detail groups.

Refer [Filter data](#) section to add or remove filters in report.

Sorting

Supports adding sort expressions to chart category and series groups and pivot table groups. It controls the order in which data is displayed in a data region item of the report (either as ascending or descending).

Refer [Sort data](#) section to control the order the data either in ascending or descending.

Grouping

User can add grouping to display data in a hierarchy view by organizing nested, adjacent, and recursive hierarchy groups. You can use a dataset field, expression or other values to group data.

Refer [Group data](#) section display data in a hierarchy view.

Drill through reports

An action property is available to specify the drill through report path. It allows end users to click on a data value in the report to view related data information in the child report. A separate child report can have detail/summary data. Also, users can add parameters to filter data based on user selection.

Refer [Report linking](#) section to specify the drill through report path.

Hyperlink

Provides property options to set hyperlink actions to text boxes, images, charts, and tablix. The hyperlink can contain data field, static or dynamic URL value expression.

Refer [Hyperlink](#) section to set hyperlink actions to text boxes, images, charts, and tablix.

Drilldown action

Drilldown action can be used to create an interactive report which allows user to enable expand-collapse action for tablix in the report.

Refer [Drilldown action](#) section to handle drilldown properties in designer.

Expression builder

Report Designer provides an expression builder that allows you to create simple and complex RDL expressions.

Refer [Expressions](#) section to create expressions in expression builder.

Format data

You can format the data to represent it in various forms that includes Numbers, Currency, Date, Time, Scientific, Percentage and Custom format.

Refer [Format data](#) section to handle formatting for data.

Unit switcher

You can switch the unit type in the report, to convert from one unit type to another.

Refer [Unit switcher](#) section to switch the unit type in the report.

Code module

You can embed a custom code or custom assemblies to a report or add references to custom code embedded in a report or generated with custom assemblies.

Refer [Code module](#) section to add and handle custom code in report.

Toolbar

The Report Designer toolbar contains a set of icons or buttons that allows you to perform common report designing operations.

![toolbar items](/static/assets/on-premise/images/report-designer/compose-report/toolbar/toolbar-menu.png)

New

To create a new blank report, click on the **New** icon in the toolbar. Now, **New Report** dialog will open as shown in below snap. In **File Name** field, specify the report name and click on the **Create** button.

| | |
|---------|------|
| Toolbar | Open |
|---------|------|

![New report dialog](/static/assets/on-premise/images/report-designer/compose-report/toolbar/new-report.png)

Open

Opens the existing report from device or server. Refer [Open report](#) section to open a report.

Save

Saves the report to device or server. Refer [Save report](#) section to save the report.

Cut

Cut action removes the currently selected report item and it can be pasted anywhere in the designer surface.

Copy

Copy action makes another copy of selected report item and it can be pasted anywhere in the designer surface.

Paste

Paste action can be used to paste the cut or copied report item in the designer surface. **Paste** will enable only if we perform **Cut** or **Copy** operation.

Delete

Delete option removes the currently selected report item in the designer surface.

Undo

Undo action records the recent actions such as add, delete, insert, properties change and other designing actions done in a report.

Redo

Redo action reverses the last undo action done in a report.

Zoom out

Zoom out option decreases the document's current zoom factor.

Zoom in

Zoom in option increases the document's current zoom factor.

Layout ordering

Layout ordering can be used to change the layout order of report items in design area surface. Refer [Layout Ordering](#) section to know about ordering modes.

Alignment

[Alignment](#) has set of alignment options that enables you to align the selected report items in the designer surface.

Distribute

[Distribute](#) option can be used to place the selected report items on the design surface at equal intervals from each other.

Sizing

[Sizing](#) can be used to equally size selected report items on the design surface.

Design surface

Align to grid

Align to grid

Snaps the top left of the selected report items to the closest gridline.

Size to grid

Snaps the selected report item to the closest gridline by resizing the report item on all four sides.

View

View menu contains options to show or hide Header, Footer, Grid Lines, Snap To Shape in the report design.

![View Menu](/static/assets/on-premise/images/report-designer/compose-report/toolbar/view-menu.png)

Header : Enables/Disables header area in the report. Refer [Header](#) section to add and remove header in designer surface.

Footer : Enables/Disables footer area in the report. Refer [Footer](#) section to add and remove footer in designer surface.

Grid Lines : Gridlines are the pattern of lines drawn behind the report items. It provides a visual guidance while dragging or arranging the objects on the designer surface. Click on the Grid Lines option in the view menu to show or hide the gridlines in the report.

Snap To Shape : When you drag a report item in design surface, snapping guidelines will indicate the alignment position to easily align with closest report item. Click on the Snap To Shape option in the view menu to enable or disable the snapping guidelines in the report design.

Preview

After designing a report, click on the Preview button to preview the report. Refer [Preview](#) section to preview the designed report.

Design surface

The Bold Report Designer comes with a WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

Key features

- Easily arrange report items on the Report Designer surface using simple drag-and-drop operations.
- Grid lines and snap-to-grid options simplifies the positioning and aligning of report items.
- You can resize a report item in eight different directions to keep its shape. You can also resize multiple objects at the same time.
- When multiple report items overlaps in the design surface, the layout ordering controls which report item is at the top and which is at the bottom.
- Enhanced alignment options that allows perfect positioning, sizing, and aligning of report items in a report.
- You can distribute selected report items on the design surface at equal intervals from each other.
- You can perform basic designing actions easily using design surface context menu.

See also[Report item resizing](#)[Report item selection](#)[Report item alignment](#)[Distribute report items](#)[Layout ordering](#)[Context Menu](#)

Report item resizing

To improve the report readability, we can resize the height and width of the report items in design area.

Resize using resizer

Resize height:

To change the height of the report items , place the mouse pointer in the **CenterTop** or **CenterBottom** position of the specific report item selection line.

Use **CenterBottom** to increase height towards downward direction or to decrease height towards upward direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/resize-vertically-downwards.png)

Use **CenterTop** to increase height towards upward direction or to decrease height towards downward direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/resize-vertically-upwards.png)

Resize width:

To change the width of the report item , place the mouse pointer in the **RightCenter** or **LeftCenter** position of the report item selection line.

Use **RightCenter** to resize the width outwards or inwards in right side direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/resize-right-center.png)

Use **LeftCenter** to resize the width outwards or inwards in left direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/resize-left-center.png)

Change width and height proportionally:

1. Select the report item in the design area.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/select-report-item-to-resize.png)

2. To resize the report item place the mouse pointer in **LeftTop** or **RightTop** or **LeftBottom** or **RightBottom** position on the report item.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/resize-proportionally.png)

3. Now, the resizer arrow will be enabled in the respective position. Hold and drag the resizer arrow, now the report item will be resized proportionally in all direction.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/over-all-resize-output.png)

Resize using width and height properties

1. Select a report item, now the respective item properties will be listed in the properties panel.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/select-report-item-to-resize.png)

2. In the properties panel, modify the **Height** and **Width** property of the specific report item.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/width-and-height-properties.png)

Resize using keyboard shortcuts

You can resize report item in design surface using the keyboard short-cut keys. Select the report item and use any of the below key combinations to resize the report item.

- To increase the width of the report item, hold **Ctrl + Shift** keys or only **Shift** and press **Right Arrow** key.
- To decrease the width of the report item use hold **Ctrl + Shift** keys or only **Shift** and press **Left Arrow** key.
- To increase the height of the report item use hold **Ctrl + Shift** keys or only **Shift** and press **Up Arrow** key.
- To decrease the height of the report item use hold **Ctrl + Shift** keys or only **Shift** and press **Down Arrow** key.

Ctrl + Shift + Arrow Keys combination increase or decreases the value by 1 point, whereas **Shift + Arrow Keys** combination increases or decreases the value by 8 points.

To know more about supported keyboard shortcut keys, refer [Design report using keyboard shortcuts](#) section.

Using touch resizer

To increase or decrease the width and height using touch resizer, select the report item and hold any of the highlighted points in below snap.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-resizing/touch-resizer.png)

Then drag towards the required direction in design area.

Report item selection

Report item selection plays major role in positioning and formatting report items in the design surface. Different ways to select report items in design surface are explained below.

Using mouse action

Mouse hover on the report item which you want select.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-selection/mouse-hover-on-the-report-item.png)

Then tap on the left or right mouse button. Now, the selection will be applied to the specific report item.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-selection/selection-output.png)

To select multiple report items in the design surface,

Hold mouse button and draw a bounding rectangle around the required item.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-selection/draw-bouding-rectangle.png)

Then release the mouse button, now the selection will be applied on those report items in the designer surface.

![Resize the table column](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-selection/multiple-selection-using-mouse.png)

Using keyboard shortcuts

- To select all report items in the design surface, hold **Ctrl** and press **A** key.
- To navigate selection between report items in design surface, use **Shift + Tab** or **Tab** keys.
- To remove selection from report items, press **Esc** key.

Report item alignment

Bold Report Designer provides a set of interactive alignment options to improve the report editing experience. The alignment options are listed in the toolbar as shown below.

![Alignment options](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/alignment-options.png)

Align

You can align the report items in design surface in left, center, right, top, middle and bottom directions. To enable this options in the toolbar, you must select minimum two report items in the design surface.

![Alignment options](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/align-options.png)

| | |
|-----------------------|-------|
| Report item alignment | Align |
|-----------------------|-------|

Left align

The selected items are aligned to left position based on the report item which has left-most position value among the selected report items.

Before aligning:

![Before left align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-left-align.png)

After aligning:

![After left align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-left-align.png)

Center align

The selected report items are aligned to center position based on the left-most and right-most positions of the selected report items.

Before aligning:

![Before left align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-left-align.png)

After aligning:

![After left align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-center-align.png)

Right align

The selected items are aligned to right position based on the report item which has right-most position value among the selected report items.

Before aligning:

![Before right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-left-align.png)

After aligning:

![After right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-right-align.png)

Top align

The selected items are aligned to top position based on the report item which has top-most position value among the selected report items.

Before aligning:

![Before right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-left-align.png)

After aligning:

![After right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-top-align.png)

| | |
|-----------------------|-------|
| Report item alignment | Align |
|-----------------------|-------|

Middle align

The selected report items are aligned to middle position based on the top-most and bottom-most positions of the selected report items.

Before aligning:

![Before left align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-left-align.png)

After aligning:

![After left align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-middle-alignment.png)

Bottom align

The selected items are aligned to bottom position based on the report item which has bottom-most position value among the selected report items.

Before aligning:

![Before right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-left-align.png)

After aligning:

![After right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-bottom-align.png)

Center horizontally

Aligns the selected report items to the center position of the design surface or parent container horizontally.

Before aligning:

![Before right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-center-horizontal-align.png)

After aligning:

![After right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-center-horizontal-align.png)

Center vertically

Aligns the selected report items to the center position of the design surface or parent container vertically.

Before aligning:

![Before right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-center-vertical-align.png)

After aligning:

![After right align](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-center-vertical-align.png)

Distribute

The Distribute option can be used to place the selected report items with equal spacing in both horizontal and vertical direction.

![Distribute Report Items](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/distribution.png)

Horizontally

The horizontal option distributes the report items in the design surface with equal intervals horizontally.

Before aligning:

![Before horizontal distribution](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-distribution.png)

After aligning:

![After horizontal distribution](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-horizontal-distribution.png)

Vertically

The vertical option distributes the report items in the design surface with equal intervals vertically.

Before aligning:

![Before vertical distribution](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/before-vertical-distribution.png)

After aligning:

![After vertical distribution](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/after-vertical-distribution.png)

Sizing

Sizing option can be used to provide equal sizing for the selected report items in the design surface.

![Sizing](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/sizing-report-item.png)

Same size

Resizes the size (width and height) of the selected report items to the size of the first selected report item.

Same height

Resizes the height of the selected report items to the height of the first selected report item.

Same width

Resizes the width of the selected report items to the width of the first selected report item.

Align to grid

Snaps the top left of the selected report items to the closest grid line in the design surface.

![Align to grid](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/align-to-grid.png)

Size to grid

Snaps the selected report item to the closest gridline by resizing the report item on all four sides.

![Size to grid](/static/assets/on-premise/images/report-designer/compose-report/design-surface/report-item-alignment/size-to-grid.png)

Context menu

A context menu is a pop-up menu that shows the basic options used in design surface. When you perform the right click action in design surface, the context menu will display options depending on the target design surface or report item.

Header and footer

To add or remove header and footer using context menu, right click on the design surface. Based on, the header and footer visibility context menu will display the following options:

1. Add Header
2. Add Footer
3. Remove Header
4. Remove Footer

Add Header

Right click on the design surface and click on **Add Header** option in the context menu.

![Add Header](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/add-header.png)

Now, header will be enabled in the report design.

Add Footer

Right click on the design surface and click on **Add Footer** option in the context menu.

![Add Footer](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/add-footer.png)

Now, footer will be enabled in the report design.

Remove Header

Right click on the design surface and click on **Remove Header** option in the context menu.

![Remove Header](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/remove-header.png)

Now, header will be removed from the report design.

Remove Footer

Right click on the design surface and click on **Remove Footer** option in the context menu.

![Remove Footer](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/remove-footer.png)

Now, footer will be removed from the report design.

Insert

Insert item at design surface

Right click on the design surface. Based on the target surface (body, footer, or header), the context menu will show the report items list in the **Insert** option.

| | |
|--------------|-----|
| Context menu | Cut |
|--------------|-----|

![Insert report items](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/insert-option.png)

Click on the required report item in the insert menu, will add the report item into the report design.

![Insert report item in design surface](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/insert-item-in-design-surface.png)

Insert item at rectangle container

Select the rectangle report item and right click on it.

![Insert item in rectangle](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/insert-item-in-rectangle.png)

Then, click on the required report item type in the insert menu. Now, the the report item will be added into the rectangle.

![Insert item in rectangle](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/insert-item-in-rectangle-demo.png)

Insert item at tablix cell

Select the cell in tablix report item and right click on it.

![Insert item in table cell](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/insert-item-in-cell.png)

Then, click on the required report item type in the insert menu. Now, the the report item will be added into the target cell.

![Insert item in table cell](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/insert-item-in-table-demo.png)

If a table cell contains rectangle, then insert action will add the new report item as child of rectangle.

Cut

Cut item from design surface

Select the required report items in design surface and right click.

![Cut from design surface](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/cut-from-design-surface.png)

Then, click on Cut option in the context menu. Now, the selected report items will be removed from report design.

Cut item from rectangle container

Select the required report items in target rectangle report item and right click.

![Cut from rectangle](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/cut-from-rectangle.png)

Then, click on Cut option in the context menu. Now, the selected report items will be removed from target rectangle.

Cut item from tablix cell

Select the cell in tablix report item and right click on it.

| | |
|--------------|----------------------|
| Context menu | Copy |
|--------------|----------------------|

![Insert item in table cell](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/cut-from-table-cell.png)

Then, click on **Cut** option in the context menu. Now, the report item will be removed from the target cell.

Select the cell in nested tablix, now the gripper for respective tablix will be enabled like below,

![Select nested tablix](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/select-nested-tablix.png)

Right click on the common gripper area,

![Common gripper area](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/common-gripper.png)

Then, click on **Cut** option in the context menu. Now, the respective nested tablix will be removed from the parent tablix cell.

[Copy](#)

[Copy item from design surface](#)

Select the required report items in design surface and right click.

![Copy from design surface](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/copy-from-design-surface.png)

Then, click on **Copy** option in the context menu. Now, the selected report items will be copied in the internal clipboard.

[Copy item from rectangle container](#)

Select the required report items in target rectangle report item and right click.

![Copy from rectangle](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/copy-from-rectangle.png)

Then, click on **Copy** option in the context menu. Now, the selected report items will be copied in the internal clipboard.

[Copy item from tablix cell](#)

Select the cell in tablix report item and right click on it.

![Copy item in table cell](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/copy-from-table-cell.png)

Then, click on **Copy** option in the context menu. Now, the selected report item will be copied in the internal clipboard.

Select the cell in nested tablix, now the gripper for respective tablix will be enabled like below,

![Select nested tablix](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/select-nested-tablix.png)

Right click on the common gripper area,

![Common gripper area](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/copy-nested-tablix.png)

Then, click on **Copy** option in the context menu. Now, the respective nested tablix will be copied in the internal clipboard.

Paste

Paste item at design area

Right click on the design surface and click on **Paste** option in the context menu. Last copied item will be pasted in the target mouse position.

![Paste in design surface](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/paste-in-design-surface.png)

Paste item at rectangle

Right click on the rectangle report item and click on **Paste** option in the context menu. Last copied item will be pasted in the rectangle based on mouse position.

![Paste in rectangle](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/paste-in-rectangle.png)

Paste item at tablix cell

Right click on the required tablix cell and click on **Paste** option in the context menu. Last copied item will be pasted in the tablix cell.

![Paste in tablix cell](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/paste-in-tablix.png)

If tablix cell contains rectangle, then copied item will be pasted inside the rectangle.

Delete

Select a report item from design surface or rectangle or tablix, then right click on the report item. Now, click on **Delete** option in the context menu. Selected report item will be removed from the report design.

![Delete report item](/static/assets/on-premise/images/report-designer/compose-report/design-surface/context-menu/delete-item.png)

Properties Panel

The Properties panel allows you to view and edit the properties of the selected report item or the entire report. It appears on the right side in the Web Report Designer.

Open properties panel

To open the properties panel, click on the **Properties** icon in the configuration panel.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/properties-icon.png)

Now, the properties panel will be displayed like below.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/initial-properties-view.png)

By default, the properties panel will display the **Report** properties.

Refer [Report properties](#) section to set and edit the report properties.

Header properties

To open the **Header** properties, focus the mouse pointer on the header area of the report.

![Open header properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/header-properties.png)

Refer [Header properties](#) section to set and edit the report header properties.

Refer [Show or hide header](#) section to show or hide header in the report.

Body properties

To open the **Body** properties, focus the mouse pointer on the body area of the report.

![Open body properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/open-body-properties.png)

Refer [Body properties](#) section to set and edit the report body properties.

Footer properties

To open the **Footer** properties, focus the mouse pointer on the footer area of the report.

![Open footer properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/open-footer-properties.png)

Refer [Footer properties](#) section to set and edit the report footer properties.

Refer [Show or hide footer](#) section to show or hide footer in the report.

Report item properties

To open any of the report item properties, select a report item in the design area. Now, the respective report item properties will be listed in the properties panel. In the below snap, a line report item is selected in the design area and the line properties are listed in the properties panel.

![Open a report item properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/open-report-item-properties.png)

Common properties

Some of the common properties like name, border, background etc are applicable for most of the report items and report layout, such properties are described under [Common properties](#) section.

Set expression

1. Click on the square icon at the right corner of the respective property.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/expression-menu.png)

2. Click on **Expression** menu to open the expression builder.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/expression-dialog.png)

To learn more about handling expressions in report designer refer [Expressions](#) section.

3. The square icon will be indicated in **Black color**, if the expression is applied to the specific property.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/set-expression-indication.png)

Reset expression

1. Click on the square icon at the right corner of the respective property.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/expression-menu.png)

2. Click on **Reset** menu option to remove the expression for the specific property.
3. The icon will be indicated in **White color**, after reset action.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/reset-expression-indication.png)

Advanced properties

Specific properties contain nested properties, which are listed under the **Advanced** property menu in the properties panel.

1. Click the square icon at the right corner of the respective property.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/advanced-menu.png)

2. Click on **Advanced...** menu option. Now, the advanced properties of the respective category will be listed like below.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/advanced-option-properties.png)

3. After customizing the properties, click on the **Close** icon to close the **Advanced Options** menu.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/close-advanced-menu.png)

Dependent properties

There are specific properties in a report items, which will be enabled in the properties view based on the value of some other property. For example when you drag and drop the rectangle item into design area, the **Page Break** property will be listed in the properties view like below.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/rectangle-page-break-property-initial-view.png)

If the **Enable Page Break** property is set to true, the dependent page break properties will be listed in the properties view like below.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/enable-page-break-property.png)

Again, if the **Enable Page Break** property is set to false, the dependent page break properties will be hidden from the properties view.

Report properties

To open the Report properties, focus the mouse pointer outside of the design area.

![Report properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/open-report-properties.png)

Basic settings

The border style, color, width and background color properties are listed under the **Basic Settings** category.

![Report basic settings](/static/assets/on-premise/images/report-designer/report-items/properties-panel/report-basic-settings.png)

Border

Refer [Border Properties](#) section.

Background color

Refer [Background color](#) section.

Background Image

Background image property can be used to display company logos, watermarks, or any other background image to the report. To set background image to the report refer [Background Image](#) section.

Code

Code Module is used to embed a custom code or assemblies to a report. To embed a custom function or assembly into the report refer [Code Module](#) section.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/custom-code-property.png)

Page Units

Page unit property is used to set the unit type for the entire report. By default, the report unit type is set to **Pixels**. Refer [Unit Switcher](#) section to switch the unit type of the report in Web Report Designer.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/page-unit-property.png)

Margin

Using the **Margin** property, you can increase or decrease the left, right, top, and bottom spacing of the report layout.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/margin-property.png)

Paper Size

The paper size property is used to specify the size of the paper, when you print the report. The paper size property determines the number of pages in a report.

Orientation

Page orientation is the direction in which a report is displayed or printed. The two basic types of page orientation are portrait (vertical) and landscape (horizontal). The default orientation is **portrait**.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/orientation-types.png)

Paper size

Select a paper size from the drop-down list to set the width and height dimensions for the report layout. This property decides the width and height of the report layout on export action.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/paper-size-types.png)

To set the custom width and height for the report layout choose **Custom** paper size. Once you select the **Custom** type in the drop-down list the **Width** and **Height** fields will be enabled, you can increase or decrease the width and height properties of the report layout.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/custom-paper-size.png)

Language

The **Language** property can be used to set the locale on a report which determines the default formats for displaying report data.

![Report Language](/static/assets/on-premise/images/report-designer/report-items/properties-panel/report-language.png)

Select the required language in the **Language** property dropdown.

The language property on a text box overrides the language property on the report.

Miscellaneous

Custom Attributes

This property can be used to add custom properties at report level. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

![Custom properties at report level](/static/assets/on-premise/images/report-designer/report-items/properties-panel/custom-properties-report-level.png)

Report Sections

Report Header and Footer

Report header and footer section allows to place the report items at the top and bottom of the report body. You can place images, text boxes, and lines, rectangles and expressions in header and footer section of the report. For example, you can place the Report Title, Company Logo, Company Address, Report Generated Time, Page Numbers etc in the headers and footers section.

Note: You cannot place data regions in the header and footer section of the report design. By default, reports have report footer, but not report header. Refer [show or hide header and footer](#) section to add or remove header and footer in a report.

Report Body

Report body section is the main work area for designing the reports. You can place all types of report items in the report body section.

General properties

Following are the general properties of report header, body and footer area.

Border

Border properties can be used to add or customize the border around header, body, or footer section in the report design. To set border properties to the report sections using properties panel, refer [Border Properties](#) section.

Background color

The background color property can be used to set the background color of the header/body/footer sections.

![Background color](/static/assets/on-premise/images/report-designer/report-items/properties-panel/body-background-color-picker.png)

Background Image

Background image property can be used to display company logos, watermarks, or any other background image to the report sections such as the page header, footer and body. To set background image to the report sections refer [Background Image](#) section.

Header and Footer properties

Height

Height property can be used to increase or decrease the height of the report header and footer area. This property is listed under the **General** category of header/footer properties in the properties panel.

![Header and footer height property](/static/assets/on-premise/images/report-designer/report-items/properties-panel/header-footer-height.png)

Print on first page

Enable this option to display header/footer only on the first page of a report instead of every page.

![Print on first page](/static/assets/on-premise/images/report-designer/report-items/properties-panel/print-on-page-property.png)

Print on last page

Enable this option to display header/footer only on the last page of a report instead of every page.

![Print on last page](/static/assets/on-premise/images/report-designer/report-items/properties-panel/print-on-last-page-property.png)

Body properties

Size

Size property can be used to set the width and height of the report body area in the report design. This property is listed under the **Position** category of body properties in the properties panel.

![Report body width and height](/static/assets/on-premise/images/report-designer/report-items/properties-panel/body-size-property.png)

The width of report header and footer will be same as the report body area.

Report Header and Footer

The header and footer can be enabled or disabled based on the report design requirements.

Show or hide report header

By default the report header will be hidden, when you create a new report in Web Report designer. To show or hide the report header, Open the **View** menu in the toolbar. Now, click on the **Header** option in the menu.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/header-icon-disabled.png)

Now, the **Header** section will be shown in the report.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/header-icon-enabled.png)

To hide the report header, again click on the **Header** option in the menu.

Show or hide report footer

By default the report footer will be shown, when you create a new report in Web Report designer. To show or hide the report footer, Open the **View** menu in the toolbar. Now, click on the **Footer** option in the menu.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/footer-icon-enabled.png)

Now, the **Footer** section will be hidden in the report.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/footer-icon-disabled.png)

To show the report footer, again click on the **Footer** option in the menu.

Common properties

This section describes the properties that are common for report layout and most of the report items in report designer.

Name

In **Name** field, type the name for the report item or use the default name. For example, the default name created for textbox report item will be similar to **TextBox1**. This is common property for all the report items.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/name-property.png)

You cannot set the name property for report, body, header and footer.

Border properties

The Border property allows you to customize the borders around report layout and report items. It also provides you with a way of setting border width, color and style for each side of report layout and report items. There are three properties of a border you can change –

- The **border-color** specifies the color of a border.
- The **border-style** specifies whether a border should be solid, dashed line, double line, or dotted.
- The **border-width** specifies the width of a border.

The border properties are listed under the **Appearance** or **Basic Settings** category in the properties panel.

![Border properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/report-basic-settings.png)

This property is not applicable for **Line** report item.

Border style

The border-style property allows you to select one of the following styles of border -

- None - No border.
- Dashed - Border is a series of short lines.
- Dotted - Border is a series of dots.
- Double - Border is two solid lines.
- Solid - Border is a single solid line.

![Border style types](/static/assets/on-premise/images/report-designer/report-items/properties-panel/border-types.png)

Border color

The border-color property is used to set colors to the border surrounding report layout or report items.

![Border color picker](/static/assets/on-premise/images/report-designer/report-items/properties-panel/border-color-picker.png)

Border width

The border-width property helps you to set the width of the border. Increase or decrease the border width in the numeric drop-down provided for border property.

![Border width](/static/assets/on-premise/images/report-designer/report-items/properties-panel/border-width-drop-down.png)

Setting borders for each side

You can individually set the border properties of the bottom, left, top and right sides of report layout or report items. The individual border options are provided under the **Advanced Properties** menu. Refer [Advanced Properties](#) section to open/close advanced properties menu.

![Border advanced property](/static/assets/on-premise/images/report-designer/report-items/properties-panel/border-advanced-properties.png)

| | |
|-------------------|------------------|
| Common properties | Background color |
|-------------------|------------------|

Set border properties based on dynamic value

You can assign expressions to set the border properties based on dynamic values. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Border expression property](/static/assets/on-premise/images/report-designer/report-items/properties-panel/border-expression-properties.png)

Background color

The background color property can be used to set the background color of the report items or report. It is listed under the **Appearance** or **Basic Settings** category in the properties panel.

![Background color property](/static/assets/on-premise/images/report-designer/report-items/properties-panel/back-ground-color-picker.png)

This property is not applicable for **Line** report item.

You can assign expressions to set the background property based on dynamic values. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Background expression property](/static/assets/on-premise/images/report-designer/report-items/properties-panel/back-ground-color-expression-menu.png)

Visibility

Enable or disable the visibility property to show or hide the report items on report preview or export action.

![Visibility property](/static/assets/on-premise/images/report-designer/report-items/textbox/visibility-property.png)

- Enable the checkbox to display the report item.
- Disable the checkbox to hide the report item.

You can also set the visibility of report items based on dynamic values using expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Visibility expression property](/static/assets/on-premise/images/report-designer/report-items/properties-panel/visibility-expression-property.png)

This property is applicable only for report items.

Position

To position and size the report items in the report design, the left, top, width and height properties are used. Those properties are listed under **Position** category in the properties panel.

![Position property](/static/assets/on-premise/images/report-designer/report-items/rectangle/position-property.png)

You can increase or decrease the values in the numeric drop-down of the respective property.

Custom properties

Custom properties can be used to set the values for additional properties of report items or whole report that are not available in the properties pane. It is listed under the **Miscellaneous** category in the properties panel.

![Custom properties](/static/assets/on-premise/images/report-designer/report-items/properties-panel/custom-properties.png)

Click on the Set Attributes... button to open Custom Attributes dialog.

![Custom properties dialog](/static/assets/on-premise/images/report-designer/report-items/properties-panel/custom-properties-dialog.png)

Click on the ADD button to define value for custom property. In the Name field provide property name and in value field assign the values.

![Assign custom property value](/static/assets/on-premise/images/report-designer/report-items/properties-panel/assign-custom-prop-value.png)

Tooltip

You can configure a tooltip for report items using the ToolTip property, it is listed under the Miscellaneous category in the properties panel.

![Report item tooltip](/static/assets/on-premise/images/report-designer/report-items/properties-panel/tooltip.png)

It can be used to display information, such as descriptive text or data related to the specific report item. When you hover over the report item in a rendered report, information will be shown in the tooltip.

To configure tooltip for a report item, select the report item in design area. In properties panel set the tooltip value as shown below,

![Set value for tooltip](/static/assets/on-premise/images/report-designer/report-items/properties-panel/set-tooltip.png)

You can also set the tooltip of report items based on dynamic values using expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Set dynamic value for tooltip](/static/assets/on-premise/images/report-designer/report-items/properties-panel/tooltip-expression.png)

This property is applicable only for report items.

Report Item Multiselection

Report item multiselection allows to set/edit the properties of report items in a easier way. Customizing more than one report item properties in a single action, makes the report designing steps as a simpler and work saving process.

Open properties panel

To open the properties panel, click on the Properties icon in the configuration panel.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/properties-icon.png)

Now, the properties panel will be displayed like below.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/properties-panel/initial-properties-view.png)

By default, the properties panel will display the Report properties.

Same item type selection

When the report items of same type is selected in the design area, the properties of the respective report item type will be listed in the properties panel. Drag and drop more than one rectangle report items in the design area and select all rectangle report items. Now, the properties of **Rectangle** report item will be listed in the properties panel.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/same-report-item-selection.png)

When you edit the report item properties on multiple report item selection, the respective property change will be applied to all selected items in the design area in single property change action. For example, if you change the **Background** color in the properties panel,

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/set-property-same-report-item-selection.png)

the background color will be applied to the selected rectangle report items in the design area.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/background-property-output.png)

Similarly, you can set other properties for the selected report items in the design area. On undo and redo action the property values will be restored in all previously selected report items.

The **Name** property of the report item will be in disabled state and indicates the selected report item type. For example, in the above snap two rectangle report items are selected and the type is indicated as **Rectangle** in the name property field. The **Name** property change is not allowed for multiple report item selection.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/name-property-for-same-type-selection.png)

Different item type selection

When the report items of different type is selected in the design area, the static properties such as border, background, position and visibility properties will be listed in the properties panel. Drag and drop more than one report items of different types in the design area and select all report items. Now, the static properties will be listed in the properties panel.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/different-report-item-selection.png)

When you edit the report item properties on multiple report item selection, the respective property change will be applied to all selected items in the design area in single property change action. For example, if you change the **Background** color in the properties panel,

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/set-property-different-report-item-selection.png)

the background color will be applied to the selected report items in the design area.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/background-property-different-item-selection.png)

Similarly, you can set other properties for the selected report items in the design area. On undo and redo action the property values will be restored in all previously selected report items.

Note: On property change action, if any of the selected report item has no support for the specific property will be skipped. For example, The line report item has no **Background Color** property support. If background property is changed for line and rectangle report item combination, the background property change will be skipped for line report item and only applied to rectangle.

The **Name** property of the report item will be in disabled state and for different report item selection it is indicated as **Common Properties**. The **Name** property change is not allowed for multiple report item selection.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/name-property-for-different-type-selection.png)

Cell with same item type

When the tablix cell containing same report item type is selected in the tablix data region, the properties of the respective report item type will be listed in the properties panel. Select multiple tablix cell containing textbox report item, now the properties of **textbox** report item will be listed in the properties panel.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/select-tablix-cell-with-same-report-item.png)

When you edit the tablix cell properties on multiple cell selection, the respective property change will be applied to all selected cells in the tablix in single property change action. For example, if you change the **Content** property for textboxes in the properties panel,

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/set-tablix-cell-property-for-same-report-item-type.png)

the content property value will be applied to selected cells in the tablix data region.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/set-content-for-textbox.png)

Similarly, you can set other properties for the selected tablix cell in the data region. On undo and redo action the property values will be restored in all previously selected tablix cells.

Cell with different item type

When the tablix cell containing different report item type is selected in the tablix data region, the static properties such as border, background, position and visibility properties will be listed in the properties panel. Select multiple tablix cell containing different report items type, now the static properties will be listed in the properties panel.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/select-tablix-cell-with-different-report-item.png)

When you edit the tablix cell properties on multiple cell selection, the respective property change will be applied to all selected cells in the tablix on a single property change action. For example, if you change the **Background** color in the properties panel,

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/set-property-different-cell-selection.png)

the background color will be applied to the selected cells in the tablix data region.

![Same report item selection](/static/assets/on-premise/images/report-designer/report-items/report-item-multi-selection/set-property-tablix-cell-ouput.png)

Similarly, you can set other properties for the selected tablix cell in the data region. On undo and redo action the property values will be restored in all previously selected tablix cells.

Sorting

Sorting arrange the data in either ascending or descending order to present data in more readable formats. It can be used to sort the data in a data regions.

![open-sort-dialog](/static/assets/on-premise/images/report-designer/compose-report/sort-data/sort-dialog.png)

Add Sorting

1. To add a sort expression, Click on the **Add** icon.

![choose-sort-fields](/static/assets/on-premise/images/report-designer/compose-report/sort-data/sort-to-choose-fields.png)

2. Dataset fields are listed in the first drop-down list, choose the necessary field from the drop-down list or set an [Expression](#).

![open-expression](/static/assets/on-premise/images/report-designer/compose-report/sort-data/expression-field.png)

3. Sort **Direction** types are listed in the second drop-down list, choose a sort direction in the drop-down list.

- o **Ascending** sorts the data in A-Z order.
- o **Descending** sorts the data in Z-A order.

![add-field-to-sort](/static/assets/on-premise/images/report-designer/compose-report/sort-data/sort-add-field.png)

4. Click OK.
5. To add multiple sort expression, follow steps 1 - 4.

![addfield multiple sort](/static/assets/on-premise/images/report-designer/compose-report/sort-data/multiple-sort.png)

Set Expression

Follow steps 1 - 4, to add sort expression.

1. To edit/create an expression, click on the square icon and select [Expression](#).

![expression-icon](/static/assets/on-premise/images/report-designer/compose-report/sort-data/sort-expression-icon.png)

2. Expression dialog will be open as shown below, Refer [Expression](#) section for better understanding with the following sections.

![open-expression](/static/assets/on-premise/images/report-designer/compose-report/filter-data/expression-dialog.png)

3. The icon will be indicated in Black color, if the expression is applied to the dataset field.

![set-expression](/static/assets/on-premise/images/report-designer/compose-report/sort-data/expression-set-black.png)

Reset Expression

1. To reset an expression, click on the square icon and select Reset.

![Select reset expression](/static/assets/on-premise/images/report-designer/compose-report/sort-data/sort-reset.png)

2. The icon will be indicated in White color, after reset action.

![After reset expression](/static/assets/on-premise/images/report-designer/compose-report/sort-data/after-reset-expression.png)

Reordering

To change the order of an sort expression, click and hold the icon in the left corner, and then drag the sort expression to higher or lower position.

![reorder fields in sort](/static/assets/on-premise/images/report-designer/compose-report/sort-data/reorder-before.png)

The position of dragged sort expression is shown as below:

![after reorder fields in sort](/static/assets/on-premise/images/report-designer/compose-report/sort-data/after-reorder.png)

Remove Sorting

Click Delete icon in the right corner to remove the respective sort expression.

![delete-fields-sorting](/static/assets/on-premise/images/report-designer/compose-report/sort-data/delete-field.png)

Grouping

Grouping can be used to organize and group data in data regions by a field value or expression value.

![open-group-dialog](/static/assets/on-premise/images/report-designer/compose-report/group-data/group-dialog.png)

Add Grouping

1. To add a group expression, Click on the Add icon.

![choose-group-fields](/static/assets/on-premise/images/report-designer/compose-report/group-data/group-to-choose-fields.png)

2. Based on the dataset assigned to the data region, dataset fields will be listed in this drop-down or else click on the square icon to set an [Expression](#).

![open-expression](/static/assets/on-premise/images/report-designer/compose-report/sort-data/expression-field.png)

3. Click OK.
4. To add multiple group expression, follow steps 1 - 3.

![addfield multiple group](/static/assets/on-premise/images/report-designer/compose-report/group-data/multiple-sort.png)

[Set Expression](#)

Follow steps 1 - 3, to add group expression.

1. To edit/create an expression, click on the square icon and select [Expression](#).

![expression-icon](/static/assets/on-premise/images/report-designer/compose-report/group-data/group-expression-icon.png)

2. [Expression](#) dialog will be open as shown below, Refer [Expression](#) section for better understanding with the following sections.

![open-expression](/static/assets/on-premise/images/report-designer/compose-report/group-data/expression-dialog.png)

3. The icon will be indicated in **Black color**, if the expression is applied to the dataset field.

![set-expression](/static/assets/on-premise/images/report-designer/compose-report/group-data/expression-set-black.png)

[Reset Expression](#)

1. To reset an expression, click on the square icon and select [Reset](#).

![Select reset expression](/static/assets/on-premise/images/report-designer/compose-report/group-data/group-reset.png)

2. The icon will be indicated in **White color**, after reset action.

![After reset expression](/static/assets/on-premise/images/report-designer/compose-report/group-data/after-reset-expression.png)

Reordering

To change the order of an group expression, click and hold the icon in the left corner, and then drag the group expression to higher or lower position.

![reorder fields in group](/static/assets/on-premise/images/report-designer/compose-report/group-data/reorder-before.png)

The position of dragged group expression is shown as below:

![after reorder fields in group](/static/assets/on-premise/images/report-designer/compose-report/group-data/group-data/after-reorder.png)

Remove Grouping

Click **Delete** icon in the right corner to remove the respective group expression.

![delete-fields-grouping](/static/assets/on-premise/images/report-designer/compose-report/group-data/expression-set-black.png)

Filters

Filters are used to filter the dataset field by passing the specific dataset field value to limit the data in a report. It can be used to filter the data in a data regions.

![open-filter-dialog](/static/assets/on-premise/images/report-designer/compose-report/filter-data/filters-dialog.png)

Add filters

1. To add a filter, Click on the **Add** icon.

![add-field-filters](/static/assets/on-premise/images/report-designer/compose-report/filter-data/filters-add.png)

2. Dataset fields are listed in the first drop-down list, choose the necessary field from the drop-down list or set an [Expression](#).

![add-expression](/static/assets/on-premise/images/report-designer/compose-report/filter-data/expression-field.png)

3. **Operator** types are listed in the second drop-down list.

![operators-in- filters](/static/assets/on-premise/images/report-designer/compose-report/filter-data/operators.png)

To filter specific range of data use **Between** operator.

![operator-between](/static/assets/on-premise/images/report-designer/compose-report/filter-data/between-operator.png)

4. In the **Value**, enter the value directly or set an [Expression](#).
5. To add multiple filters, follow steps 1 - 4.

![add-with-multiple-filter](/static/assets/on-premise/images/report-designer/compose-report/filter-data/multiple-filters.png)

6. Click **OK**.

Set Expression

Follow steps 1 - 4, to add filters.

1. To edit/create an expression, click on the square icon and select **Expression**.

![expression-icon-shown](/static/assets/on-premise/images/report-designer/compose-report/filter-data/expression-icon.png)

2. **Expression** dialog will be open as shown below, Refer [Expression](#) section for better understanding with the following sections.

![open-expression-dialog](/static/assets/on-premise/images/report-designer/compose-report/filter-data/expression-dialog.png)

3. The icon will be indicated in **Black color**, if the expression is applied to the dataset field.

![set-expression](/static/assets/on-premise/images/report-designer/compose-report/filter-data/expression-set-black.png)

Reset Expression

1. To reset an expression, click on the square icon and select **Reset**.

![Select filter reset expression](/static/assets/on-premise/images/report-designer/compose-report/filter-data/expression-icon-reset.png)

2. The icon will be indicated in **White color**, after reset action.

![After reset expression](/static/assets/on-premise/images/report-designer/compose-report/filter-data/after-reset-expression.png)

Reordering

To change the order of an filter, click and hold the icon in the left corner, and then drag filters to higher or lower position.

![reorder fields in sort](/static/assets/on-premise/images/report-designer/compose-report/filter-data/reorder-before.png)

The position of dragged sort expression is shown as below:

![after reorder fields in sort](/static/assets/on-premise/images/report-designer/compose-report/filter-data/after-reorder.png)

Remove Filters

Click **Delete** icon in the right corner to remove the respective filters.

![delete-filter](/static/assets/on-premise/images/report-designer/compose-report/filter-data/delete-a-filter.png)

Format

Format is used to represent the text in various forms that includes **Numbers**, **Currency**, **Date**, **Time**, **Scientific**, **Percentage** and **Custom** format .

![open-format-dialog](/static/assets/on-premise/images/report-designer/compose-report/format-data/formatdialog.png)

Format Numbers

Number format is used to format the numeric values present in a textbox.

Number type has the following options:

- **Decimal Places** is used to round off the number of decimal digits in numeric values.
- **Negative Values** is used to specify the representation of negative numbers.
- **Representation** is used to display the value in Thousands, Millions, or Billions.
- **Thousand Separator** is used to separate the value in terms of thousands in number positions.

![number-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-numbers.png)

- **Use Regional Formatting** : Enable this option to apply default local culture settings to the numeric values.

![regional-option-in-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-by-regional-option.png)

Format Currency

Currency format is used to format the currency values present in a textbox.

Currency type has the following options:

- **Decimal Places** is used to round off the number of decimal digits in numeric values.
- **Negative Values** is used to specify the representation of negative numbers.
- **Representation** is used to display value in Thousands, Millions, or Billions.
- **Thousand Separator**: is used to separate the value in terms of thousands in number positions.

For example, if the field value is 1,789,905,394 and you select Billions and specify 2 decimal places, the value displayed in the report is **1.78**.

- **Currency Culture** is used to prefix the currency symbol in field value based on the selected culture.

For example, if the field value is 1,789,905,394 and you select currency culture as **English(United States)**, the value displayed in the report is **US\$1,789,905,394**.

- **Include Space** Enable this option to include space between currency culture code and value.

For example, if the field value is 1,789,905,394 and select currency culture as English(United States) and if include space checkbox is enabled, the value displayed in the report is US\$ 1,789,905,394.

![currency-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-currency.png)

Format Date

Date format is used to format the date and time field to display only date in the textbox. The supported date formats are listed in the Date drop-down list as shown below.

![date-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-a-date.png)

Format Time

Time format is used to format the date and time field to display only time in the textbox. The supported time formats are listed in the Time drop-down list as shown below.

![time-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-a-time.png)

Format Scientific

Scientific format is used to display numbers in scientific notation. The number is transformed into a real number followed by E+n, where E (which stands for Exponent) multiplies the real number by 10 to the nth power.

![scientific-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-scientific.png)

For example, a 2-decimal scientific format displays 12345678901 as 1.23E+10, which is 1.23 times 10 to the 10th power.

Format Percentage

Percentage format is used to display the value by multiplying it with 100 and appending percent (%) symbol.

Percentage type has following options:

- **Decimal Places**: is used to round off the number of decimal digits in the numerical value.
- **Include Space**: Enable this option to include space between percent symbol and value.

![percentage-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-percentage.png)

Format Custom

Custom format is used to modify the predefined formats to create a new custom number format.

![custom-format](/static/assets/on-premise/images/report-designer/compose-report/format-data/format-by-custom.png)

Linking

Linking can be used to create an interactive report using **Hyperlink** and **Report Linking** action. This property is listed under **Link** category in properties panel.

![show-link-action](/static/assets/on-premise/images/report-designer/compose-report/link-data/enable-link-action.png)

Report Linking

Report Linking allows to access the main report from the server and retains the original report definition.

Report Path

1. Choose **Report** option in the **Link To** dropdown, now the fields required to configure report path will be enabled under the link property.

![link-fields](/static/assets/on-premise/images/report-designer/compose-report/link-data/enable-report-action.png)

2. Click on the browse button in **Report** option as shown below.

![link-fields](/static/assets/on-premise/images/report-designer/compose-report/link-data/link-reportfields.png)

3. Open any folder in the browse dialog.

![browse-from-dialog](/static/assets/on-premise/images/report-designer/compose-report/link-data/browse-report-dialog.png)

4. Select any report that are shown in the browse dialog and click **OK**.

![select report from folder](/static/assets/on-premise/images/report-designer/compose-report/link-data/select-report.png)

5. Selected report path is linked in the **Report** option as shown below.

![selected report path](/static/assets/on-premise/images/report-designer/compose-report/link-data/report-path.png)

Set Parameters

To specify parameters for **Report Linking**, follow the below steps.

1. Click on the **Set Parameters** button, **Parameters** dialog will be open as shown below.

![show-parameter-dialog](/static/assets/on-premise/images/report-designer/compose-report/link-data/enable-link-parameter-dialog.png)

2. The main report parameter names are available in the drop-down list, if the report is linked with the main report. To add **New** parameter, click on the **Add** icon.

![Add-row-in-link-parameter-dialog](/static/assets/on-premise/images/report-designer/compose-report/link-data/add-row-in-link-parameter.png)

3. In the **Parameter Name**, enter the name of the report parameter.
4. In the **Value**, enter the parameter value.

![shown-expression](/static/assets/on-premise/images/report-designer/compose-report/link-data/expression-icon.png)

5. Click on the **OK** button to save the parameters.

Set Expression

Follow steps 1 - 4, to add parameters.

1. To edit/create an expression, click on the square icon and select **Expression**.

![menu-in-expression](/static/assets/on-premise/images/report-designer/compose-report/link-data/expression-menu.png)

2. **Expression** dialog will be open as shown below, Refer [Expression](#) section for better understanding with the following sections.

![shown-expression-dialog](/static/assets/on-premise/images/report-designer/compose-report/link-data/expression-dialog.png)

3. The icon will be indicated in **Black color**, if the expression is applied to the parameter.

![set-expression](/static/assets/on-premise/images/report-designer/compose-report/link-data/set-an-expression.png)

Reset Expression

1. To reset an expression, click on the square icon and select **Reset**.

![Select reset option](/static/assets/on-premise/images/report-designer/compose-report/link-data/select-reset-option.png)

2. The icon will be indicated in **White color**, after reset action.

![After reset expression](/static/assets/on-premise/images/report-designer/compose-report/link-data/after-reset.png)

Remove Parameters

Click **Delete** icon to remove the parameters.

| | |
|------------------|-----------|
| Background image | Hyperlink |
|------------------|-----------|

![parameter-delete-action](/static/assets/on-premise/images/report-designer/compose-report/link-data/delete-parameter.png)

Hyperlink

Hyperlink is used to link to a webpages in a report. Follow the below steps to add a hyperlink.

Add a hyperlink

1. Select **URL** option in the **Link To** property dropdown, now the fields required to specify external URL will be enabled under the link property.

![enable-url-in-link](/static/assets/on-premise/images/report-designer/compose-report/link-data/enable-link-url.png)

2. In **URL** option, enter URL or set an [Expression](#) that evaluates to the URL as shown below.

![provide-the-url](/static/assets/on-premise/images/report-designer/compose-report/link-data/enable-link-url-save.png)

3. To view the link, click **Preview** and click on the report item, it will navigate to the given **URL** link.

Background image

Background image property allows you to show an image in the background of a report sections such as the page header, page footer, or report body. The source for background image can be a URL of an image on the report server, an image from a dataset field, or an embedded image.

Open the properties of report sections(page header, page footer, or report body) to which you want to add a background image. The background image properties are listed under **Background Image** category in properties panel.

![Background image property](/static/assets/on-premise/images/report-designer/compose-report/background-image/background-image-property.png)

Source

External - An image that can be accessed via a URL.

![Background image source external](/static/assets/on-premise/images/report-designer/compose-report/background-image/source-external.png)

Embedded - An image that is part of the report definition.

![Background image source embedded](/static/assets/on-premise/images/report-designer/compose-report/background-image/source-embedded.png)

Database - An image stored in a database, which is accessed through a dataset.

![Background image source database](/static/assets/on-premise/images/report-designer/compose-report/background-image/source-database.png)

Value

If **External** provide the image URL.

| | |
|------------------|-----------|
| Background image | MIME Type |
|------------------|-----------|

![External value as image](/static/assets/on-premise/images/report-designer/compose-report/background-image/external-value.png)

If **Embedded** set the embedded image name.

![Embedded value as image](/static/assets/on-premise/images/report-designer/compose-report/background-image/embedded-value.png)

If **Database** choose the dataset field that contains the image.

![Database value as image](/static/assets/on-premise/images/report-designer/compose-report/background-image/database-value.png)

MIME Type

Set the **MIME Type** property to get images from the database to work. If the **Source** property is set to **External** or **Embedded**, the value of MIME type is ignored.

![MIME property](/static/assets/on-premise/images/report-designer/compose-report/background-image/mime-type-property.png)

Background repeat

![Background repeat](/static/assets/on-premise/images/report-designer/compose-report/background-image/background-repeat.png)

Repeat: Repeats the image in both horizontal and vertical direction.

RepeatX: Repeats the image in horizontal direction.

RepeatY: Repeats the image in vertical direction.

Clip: Image appears once, anchored in top left corner.

Fit: Fits the image in container.

Add company logo as the background to a report

Open the **Body** properties in properties pane.

![Body properties](/static/assets/on-premise/images/report-designer/compose-report/background-image/body-properties.png)

Embed the required image in the report.

Set the **Source** type as **Embedded** and choose the embedded image in **Value** field. Then set the background repeat as required.

![Set Bacground for report body](/static/assets/on-premise/images/report-designer/compose-report/background-image/set-bg-image-in-body.png)

Similarly, you can set background image in report header, footer and overall report.

This is how the background image appears in Design mode:

![Background image design](/static/assets/on-premise/images/report-designer/compose-report/background-image/design-mode.png)

This is how the background image appears in Preview mode:

![Background image preview](/static/assets/on-premise/images/report-designer/compose-report/background-image/preview-mode.png)

Create SSRS drilldown report

Drilldown action can be used to create an interactive report which allows user to expand or collapse report items, rows and columns associated with a group in a table or matrix.

Create data

To present data in the table, create a dataset and bind data to the table data region. In this designing section, the following dataset query is used for dataset creation.

`sql

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
  
Sales.SalesOrderHeader SOH INNER JOIN  
  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN  
  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
  
`
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Add drilldown action to a table group

Design a [simple table report](#) with a parent group and child group as shown below.

![Table design](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/table-final-design.png)

Select **SubCat** field in the grouping panel, now the group properties will be loaded in the properties panel.

![Group properties](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/open-group-properties.png)

In **Visibility** property, set the visibility of **SubCat** group to show or hide the group when first time you preview a report.

![Visibility property](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/visibility-property.png)

- Enable the checkbox to display the group.
- Disable the checkbox to hide the group.
- To determine the visibility state at run time, set expression for visibility property.

Show group on initial display

Set the visibility for **SubCat** group as **true** in the report so that sub-category group values will be in expanded state on preview.

![Initial visible state](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/initial-visible-state.png)

In **Toggle** property drop-down list, available textbox report items in the report will be listed. Choose the **ProdCat** field textbox to expand or collapse the **SubCat** based on product categories.

![Select group to toggle](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/select-group-to-toggle.png)

The text box that you use for the toggle must be in a containing scope that controls the item that you want to show or hide.

To differentiate the values, apply background color and font styles for parent and child group.

![Improve table appearance](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/improve-table-appearance.png)

Now, click on the **Preview** button to see the report preview. Here, > symbol before each product category is the drill down action.

![SSRS Drill down report](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/preview-for-drill-down-with-group.png)

Initial Toggle state:

In the report preview notice the drilldown action symbol, the group values are in expanded state but shows collapsed symbol instead of expand symbol. So to maintain the symbol state **Initial Toggle State** property can be used.

In the **ProdCat** textbox properties change the **Initial Toggle State** as **false**.

![Initial toggle state](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/initial-toggle-state.png)

Now, click on the **Preview** button and notice the drilldown symbol is in expanded state.

![Initial toggle state](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/drill-down-action-indicator.png)

Hide group on initial display

To hide the **SubCat** group values when you preview the report, set the visibility for **SubCat** group as **false**.

![Hide group visibility on preview](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/disable-initial-visible-state.png)

Then, set the **Initial Toggle State** as **True** in the **ProdCat** textbox properties.

![Hide group visibility on preview](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/toggle-drill-down-indicator.png)

Now, click on the **Preview** button to see the report preview. Notice the sub-category group values are in collapsed state and the drilldown action symbol also is in collapsed state.

![Hide group visibility on preview](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/preview-for-drill-down-with-group-hidden.png)

To see the sub-category product and their sales value, expand the group by clicking on the > symbol.

![Hide group visibility on preview](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-down-report/preview-for-drill-down-with-single-group.png)

Create SSRS drill through report

Drill through can be used to create an interactive report which allows user to open an report by clicking the link within the main report.

Create data

To present data in the table, create a dataset and bind data to the table data region. In this designing section, the following dataset query is used for dataset creation.

`sql

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
Sales.SalesOrderHeader SOH INNER JOIN  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
`
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Design drill through report

Design the drill through report, before configuring the drill through action in the main report. Refer the [Data bar report](#) section and the report design will look like below.

![Drill through report](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/databar-report-design.png)

Now, define a [basic parameter](#) and name it as Product Category.

![Basic parameter](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/basic-parameter.png)

Then apply filter condition on the dataset to filter product category and its sub categories at dataset level.

![Dataset filter](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/open-dataset-filter.png)

Add a filter and choose **Prodcat** field.

![Add filter](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/add-filter.png)

Assign a **Product Category** parameter as expression in the value field and click **OK**.

![Add filter](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/assign-parameter.png)

Refer [Dataset Filters](#) section to apply filters at dataset level.

Add drill through action to a chart

Now, design a main report and link the drill through report using **Link** property. Here, we are going to configure the drill through action for chart series.

Refer the [Pie chart](#) section and design the report like below.

![Main report](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/main-report.png)

1. Open the chart properties,

![Chart properties](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/chart-properties.png)

2. Now, choose **Report** option in **Link To** property.

![Link to property](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/choose-report-action.png)

3. In the **Report** field, browse and set the path of already designed data bar report.

![Set report path](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/set-report-path.png)

4. Click on **Set Parameters** button. Now, the parameters dialog will open like below.

![Set parameters](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/open-parameters-dialog.png)

5. Click on **ADD** option in the dialog to configure parameter values.

![Add parameter](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/add-parameter-field.png)

6. In the **Parameter Name** field, select the parameter name of the selected report.

![Select parameter](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/select-parameter.png)

7. Provide the `=Fields!ProdCat.Value` expression value in Parameter Value field and click OK.

![Provide parameter value](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/provide-parameter-value.png)

Refer [Link Report](#) section to know more about report linking.

![Set report action](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/set-report-action.png)

Drill through chart series

Now, click on the Preview button and the report preview will look like normal pie chart.

![Main report preview](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/report-preview.png)

Then click on any series in the report preview,

![Main report preview](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/click-on-series.png)

The sub-report displays sub-category wise sales amount of Bikes product. Click on the Go to parent icon to navigate from sub-report to main report.

![Drill through report](/static/assets/on-premise/images/report-designer/compose-report/create-ssrs-drill-through-report/drill-through-report.png)

Create Multiple-Column Report

Multiple column report displays data in multiple sets of columns across the page. On report print action, prints adjacent columns until there is no free space left on a page. You cannot design a report which has a multi-column layout on the top half of the report, and a table layout on the bottom half of the report.

Properties

Open or create a table report,

![Grouping Aggregate](/static/assets/on-premise/images/report-designer/compose-report/multi-column/report-design.png)

Now, open the Report Properties in the properties panel by focusing the mouse pointer outside of the design area. Multi-column properties are listed under the Page Column category in the properties list.

![Page column properties](/static/assets/on-premise/images/report-designer/compose-report/multi-column/multi-column-properties.png)

In Columns property, provide the number of columns in the report.

In Column Spacing property, type the width of the space between columns.

Then set the required Margin and Paper Size properties.

![Set page column properties](/static/assets/on-premise/images/report-designer/compose-report/multi-column/set-properties.png)

Preview

Click on the **Preview** button, now the report preview will be displayed as a single column. To view the table in multi-column layout, click **Print Layout** option in Report Viewer toolbar.

![Print Layout](/static/assets/on-premise/images/report-designer/compose-report/multi-column/print-layout.png)

In export actions such as HTML, Excel, Word, and CSV, a multi-column layout is displayed as a single column. Only **PDF** and **Power Point** rendering formats displays data in multiple columns.

![Multi column report](/static/assets/on-premise/images/report-designer/compose-report/multi-column/multi-column-report.png)

Interactive sorting

Interactive sorting can be used to enable a user to toggle between ascending and descending order dynamically in the Report Viewer by setting interactive sorting on a **TextBox** report item within a data region.

Create data

To present data in the table, create a dataset and bind data to the table data region. In this designing section, the following dataset query is used for dataset creation.

```
`sql
```

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
  
Sales.SalesOrderHeader SOH INNER JOIN  
  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryId =  
P.ProductSubcategoryId INNER JOIN  
  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryId  
`
```

Refer [Create Data](#) section and create dataset using the above query. **AdventuresWorks** database is used here.

Add interactive sorting

Design a [simple table report](#) as shown below.

![Table design](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/table-design.png)

Select the column header textbox of the column to which you want to enable interactive sorting. Here, lets add interactive sorting on the **Product Category** column.

![Select column header](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/select-column-header.png)

In **User Sort** property, set the interactive sorting using following fields:

![Sorting properties](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/sorting-properties.png)

- **Sort Expression:** Available fields in the assigned data set will be listed in the **Sort Expression** field. Select the expression containing the value of the field on which you want to provide sorting.

![Sorting properties](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/sort-expression-field.png)

- **Sort Expression Scope:** Available details, row and column grouping in the respective data region will be listed in the **Sort Expression Scope** field.
- To enable the interactive sorting on **Details Row** then choose details group.
- To enable the interactive sorting on **Grouping Column** then choose the group.

![Sorting properties](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/sort-scope.png)

- **Sort Target:** Select the grouping level or data region within the report to sort.

![Sorting properties](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/sort-target.png)

By default sorting will be applied to the current scope, you can optionally choose a different scope in **Sort Expression Scope** and **Sort Target** fields.

Preview

On report preview, interactive sorting button will be enabled in the respective column header.

![Preview design](/static/assets/on-premise/images/report-designer/compose-report/interactive-sorting/preview-design.png)

Click on the arrow icon to sort the column dynamically in ascending or descending order.

Unit Switcher

Unit switcher allows to switch the unit type in the report, to convert from one unit type to another.

Supported Unit Types

1. The following unit types are support in reportdesigner:

- Inches
- Centimeters
- Pixels
- Points
- Millimeters

- Picas

Follow the below steps to switch unit type in a report.

2. Click on the **Properties** icon to Open **Report Properties**.

![open-report-properties](/static/assets/on-premise/images/report-designer/compose-report/unit-switcher/open-report-properties.png)

3. Under the **Page Units** category, select **Page Unit** dropdown to switch the report unit type.

![propertypanel-unitswitch](/static/assets/on-premise/images/report-designer/compose-report/unit-switcher/unitswitcher-properties.png)

4. Selected unit type will be applied to the report as shown below.

![Pixels unit type](/static/assets/on-premise/images/report-designer/compose-report/unit-switcher/unittype-in-pixels.png)

Layout Ordering

Ordering option is used to change the layout order of report items in design area surface and it's also known as z-order. It contains the following ordering modes.

- Send Backward
- Bring Forward
- Send To Back
- Bring To Front

![ordering-layout](/static/assets/on-premise/images/report-designer/compose-report/layout-ordering/layout-ordering.png)

Send Backward

Visually move the selected object behind its closest intersected object in the designer surface

![send-backward-item](/static/assets/on-premise/images/report-designer/compose-report/layout-ordering/send-backward.png)

Bring Forward

Visually move the selected object over its closest intersected object in the designer surface

![bring-the-object-forward](/static/assets/on-premise/images/report-designer/compose-report/layout-ordering/bring-forward.png)

Send To Back

Visually move the selected object behind all other intersected objects in the designer surface

![send-all-object-back](/static/assets/on-premise/images/report-designer/compose-report/layout-ordering/send-to-back.png)

Bring To Front

Visually move the selected object over all other intersected objects in the designer surface

![send-all-object-front](/static/assets/on-premise/images/report-designer/compose-report/layout-ordering/bring-to-front.png)

Expression Builder

Expressions provide great flexibility to control the content, style, data, value and behavior of your reports. Expressions must start with equal sign (=), if not the value will be considered as a string literal.

![set-an-expression](/static/assets/on-premise/images/report-designer/compose-report/expressions/set-expression.png)

1. Choose **Options** from the first drop-down list to set expression with **Built-in-fields**, **Operators** and **Functions**.

![expression-in-options](/static/assets/on-premise/images/report-designer/compose-report/expressions/options-dropdown.png)

2. Choose **Data** from the second drop-down list to set expression with **Parameters** and **Dataset Fields**.

![expression-in-data](/static/assets/on-premise/images/report-designer/compose-report/expressions/data-dropdown.png)

Supported Expressions

The supported expression types and functions details are listed in the following table:

| References | Description | Example |
|------------|-------------|---------|
| ----- | ----- | ----- |

|[Built-in-fields](#)| Built-in fields collection are the global variables that are used in a report to specify the report name, page number, execution time. It includes Globals and the User collections.|=Globals!ExecutionTime|

|[Operators](#)| An operator is a symbol that represents to perform simple basic operation. It is used to combine references in an expression.|=Globals!ExecutionTime+User!UserID|

|[Functions](#)| Functions in an expressions is used to perform some basic functions like datetime, math, aggregate, text functions, conversion functions.|=Round(1.3*5)/5|

|[Parameters](#)| Represents the collection of report parameters, each of which with a parameter value.|Parameters!ReportParameter1.Value|

|[Dataset Fields](#)| Represents dataset fields collections in reports with dataset field value.|Fields!EmployeeID.Value,"DataSet1"|

Dataset Fields in Expressions

Dataset Fields that are available in the report are listed in the **Data** dropdown list.

Displaying Fields Collection

1. Choose the **Dataset Fields** that are listed in the **data** option.

![list-of-dataset-fields](/static/assets/on-premise/images/report-designer/compose-report/expressions/dataset-fields-listed.png)

2. Fields Collections are listed below in the list view, double click on any fields collection.

![field collection in list view](/static/assets/on-premise/images/report-designer/compose-report/expressions/field-collection-list.png)

3. Dataset field values are shown as expression in the textarea and click **OK** to save the selected expression.

![field collection in list view](/static/assets/on-premise/images/report-designer/compose-report/expressions/selected-expression.png)

Parameters in Expressions

Parameters that are available in the report are listed in the **Data** dropdown list.

![list-of-parameter-dropdown](/static/assets/on-premise/images/report-designer/compose-report/expressions/parameter-dropdown.png)

Displaying Parameters Collection

1. After Report parameters creation, report parameters are listed in the list view, double click on any of the report parameter.

![parameter collection in list view](/static/assets/on-premise/images/report-designer/compose-report/expressions/choose-parameter.png)

2. Parameter field values are shown as expression in the textarea and click **OK** to save the selected expression.

![selected-parameter-textarea](/static/assets/on-premise/images/report-designer/compose-report/expressions/selected-parameter.png)

Built-in Fields

Built-in fields collection are the global variables that are used in a report to specify the report name, page number, execution time. It includes Globals and the User collections.

![set-an-expression](/static/assets/on-premise/images/report-designer/compose-report/expressions/build-in-fields.png)

Global Collection

The Globals collection provides values such as the name of the report, the time when report processing start, and current page numbers for the report header or footer.

| Member | Type | Description |
|-------------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ExecutionTime | DateTime | The date and time that the report start to run. |
| PageNumber | Integer | The current page number that can be reset through the use of page breaks. |
| ReportFolder | String | The full path to the folder containing the report. This does not include the report server URL. |
| ReportName | String | The name of the report as it is stored in the report server database. |
| ReportServerUrl | String | The URL of the report server on which the report is being run. |
| TotalPages | Integer | The total number of pages in the current continuous page sequence can be used only in the page header and footer. The number can be reset by using page breaks. |
| PageName | String | The name of the current page in the report can be used only in the page header or footer. |
| OverallPageNumber | Integer | The page number of the current page for the entire report. This value is not affected by ResetPageNumber. |
| OverallTotalPages | Integer | The total number pages for the entire report. This value is not affected by ResetPageNumber. |
| RenderFormat | RenderFormat | Information about the current rendering request. |

User Collection

The User collection provides the user identifier and language settings. These values can be used in expressions to filter results in a report.

| Member | Type | Description |
|----------|--------|---------------------------------------------------|
| Language | String | The language ID of the client running the report. |
| UserID | String | The ID of the user running the report. |

Built-in Functions

Text Functions

| References | Description | Example |
|------------|--------------------------------------------|--------------------------------|
| Asc | Returns an integer value representin g the | =Asc(Fields!Description.Value) |

| | | |
|---------|------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| | character code corresponding to a character. | |
| AscW | Returns an integer value representing the character code corresponding to a character. | =AscW(Fields!Description.Value) |
| Chr | Returns the character associated with the specified character code. | =Chr(65) |
| ChrW | Returns the character associated with the specified character code. | =ChrW(241) |
| Filter | Returns a zero-based array containing a subset of a string array based on specified filter criteria. | =Filter(Parameters!MultivalueParameter.Value, "3", True, CompareMethod.Binary) |
| Format` | Returns a formatted string according to the | =Format(Globals!ExecutionTime, "Long Date") |

| | | |
|----------------|--------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| | instructions in a format string expression. | |
| FormatCurrency | Returns an expression formatted as a currency value using the currency symbol defined in the system control panel. | =FormatCurrency(Fields!YearlyIncome.Value, 0) |
| FormatDateTime | Returns a string expression representing a date/time value. | =FormatDateTime(Fields!BirthDate.Value, DateFormat.ShortDate) |
| FormatNumber | Returns an expression formatted as a number. | =FormatNumber(Fields!Weight.Value, 2) |
| FormatPercent | Returns an expression formatted as a percentage (that is, multiplied by 100). | =FormatPercent(Fields!Sales.Value/Sum(Fields!Sales.Value, "DataSet1"), 0) |
| GetChar | Returns a char value representing the character from the specified | =GetChar(Fields!Description.Value, 5) |

| | | |
|----------|------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------|
| | index in the supplied string. | |
| InStr | Returns an integer specifying the start position of the first occurrence of one string within another. | =InStr(Fields!Description.Value, "car") |
| InStrRev | Returns the position of the first occurrence of one string within another, starting from the right side of the string. | =InStrRev(Fields!Description.Value, "car") |
| Join | Returns a string created by joining a number of substrings in an array. | =Join(Parameters!MultivalueParameter.Value,",") |
| LCase | Returns a string or character converted to lowercase. | =LCase(Fields!Description.Value) |
| Left | Returns a string containing a specified number of | =Left(Fields!Description.Value, 4) |

| | | |
|---------|-------------------------------------------------------------------------------------------------|---------------------------------------------------------|
| | characters from the left side of a string. | |
| Len` | Returns an integer containing either the number of characters in a string or the number. | =Len(Fields!Description.Value) |
| LSet | Returns a left-aligned string containing the specified string adjusted to the specified length. | =LSet(Fields!Description.Value, 4) |
| LTrim | Returns the string without left side trailing spaces in the given string. | =LTrim(Fields!Description.Value) |
| Mid | Returns a string containing a specified number of characters from a string. | =Mid(Fields!Description.Value, 3, 4) |
| Replace | Returns a string in which a specified substring | =Replace(Fields!Description.Value, "tube", "headlight") |

| | | |
|-------|--------------------------------------------------------------------------------------------------|------------------------------------------|
| | has been replaced with another. | |
| Right | Returns a string containing a specified number of characters from the right side of a string. | =Right(Fields!Description.Value, 4) |
| RSet | Returns a right-aligned string containing the specified string adjusted to the specified length. | =RSet(Fields!Description.Value, 4) |
| RTrim | Returns the string without right side trailing spaces in the given string. | =RTrim(Fields!Description.Value) |
| Space | Returns a string consisting of the specified number of spaces. | =Space(3) |
| Split | Returns a zero-based, one-dimensional | =Split(Fields!ListWithCommas.Value, ",") |

| | | |
|------------|----------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------|
| | array containing a specified number of substrings. | |
| StrComp | Returns -1, 0, or 1, based on the result of a string comparison. | =StrComp(Fields!Description.Value, First(Fields!Description.Value)) |
| StrConv | Returns a string converted as specified. | =StrConv(Fields!Description.Value, vbProperCase) |
| StrDup | Returns a string or object consisting of the specified character repeated the specified number of times. | =StrDup(3, "M") |
| StrReverse | Returns a string in which the character order of a specified string is reversed. | =StrReverse(Fields!Description.Value) |
| Trim | Returns the string without trailing spaces in the given string | =Trim(Fields!Description.Value) |

| | | |
|-------|---------------------------------------------------------------------------------------|----------------------------------|
| UCase | Returns a string or character containing the specified string converted to uppercase. | =UCase(Fields!Description.Value) |
|-------|---------------------------------------------------------------------------------------|----------------------------------|

Date Time Functions

| References | Description | Example |
|------------|---------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| CDate | Convert to date. | =CDate(Fields!BirthDate.Value) |
| DateAdd | Returns a long value specifying the number of time intervals between two date values. | =DateAdd("d", 3, Fields!BirthDate.Value) |
| DatePart | Returns an integer value containing the specified component of a given date value. | =DatePart("q", Fields!BirthDate.Value, 0, 0) |
| DateSerial | Returns a date value representing a | =DateSerial(DatePart("yyyy", Fields!BirthDate.Value)-10, DatePart("m", Fields!BirthDate.Value)+3, DatePart("d", Fields!BirthDate.Value)-1) |

| | | |
|------------|----------------------------------------------------------------------------------------------------------|--------------------------------|
| | specified year, month, and day, with the time information set to midnight (00:00:00). | |
| DateString | Returns or sets a string value representing the current date according to your system. | =DatePart("m", DateString()) |
| DateValue | Returns a date value containing the date information represented by a string, with the time information. | =DateValue("January 15, 2010") |
| Day | Returns an integer value from 1 through 31 representing the | =Day(Fields!BirthDate.Value) |

| | | |
|----------------|---------------------------------------------------------------------------------|---------------------------------------------------------------|
| | day of the month. | |
| FormatDateTime | Returns a string expression representing date/time value. | =FormatDateTime(Fields!BirthDate.Value, DateFormat.ShortDate) |
| Hour | Returns an integer value from 0 through 23 representing the hour of the day. | =Hour(Fields!BirthDate.Value) |
| Minute | Returns an integer value from 0 through 59 representing the minute of the hour. | =Minute(Fields!BirthDate.Value) |
| Month | Returns an integer value from 1 through 12 representing the month of the year. | =Month(Fields!BirthDate.Value) |

| | | |
|-----------|-------------------------------------------------------------------------------------------|----------------------------------------------------|
| MonthName | Returns a string value containing the name of the specified month. | =MonthName(10, True) |
| Now | Returns a date value containing the current date and time according to your system. | ="This time tomorrow is " & DateAdd("d", 1, Now()) |
| Second | Returns an integer value from 0 through 59 representing the second of the minute. | =Second(Fields!BirthDate.Value) |
| TimeOfDay | Returns or sets a date value containing the current time of day according to your system. | ="Time of the day is " & TimeOfDay() |

| | | |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| Timer | Returns a double value representing the number of seconds elapsed since midnight. | <pre>= "Number of seconds since midnight " & Timer()</pre> |
| TimeSerial | Returns a date value representing a specified hour, minute, and second, with the date information set relative to January 1 of the year 1. | <pre>=TimeSerial(DatePart("h", Fields!BirthDate.Value), 'DatePart("n", Fields!BirthDate.Value), DatePart("s", Fields! BirthDate.Value))</pre> |
| TimeString | Returns or sets a string value representing the current time of day according to your system. | <pre>=TimeString()</pre> |
| TimeValue | Returns a date value containing | <pre>=TimeValue(Fields!BirthDate.Value)</pre> |

| | | |
|-------------|-----------------------------------------------------------------------------------------------------------|---------------------------------------------|
| | g the time information represented by a string, with the date information set to January 1 of the year 1. | |
| Today | Returns or sets a date value containing the current date according to your system. | = "Tomorrow is " & DateAdd("d", 1, Today()) |
| Weekday | Returns an integer value containing a number that represents the day of the week. | = Weekday(Fields!BirthDate.Value, 0) |
| WeekdayName | Returns a string value containing the name of the | = WeekdayName(2, True, 0) |

| | | |
|------|---------------------------------------------------------------------|-------------------------------|
| | specified weekday. | |
| Year | Returns an integer value from 1 through 9999 representing the year. | =Year(Fields!BirthDate.Value) |

Math Functions

| References | Description | Example |
|------------|---------------------------------------------------------------------------|------------------------------------------------------------|
| Abs | Returns the absolute value of a single-precision floating-point number. | =Abs(Fields!YearlyIncome.Value - 80000) |
| Acos | Returns the angle whose cosine is the specified number. | =Acos(Fields!Angle.Value) |
| Asin | Returns the angle whose sine is the specified number. | =Asin(Fields!Angle.Value) |
| Atan | Returns the angle whose tangent is the specified number. | =Atan(Fields!Tangent.Value) |
| tan2 | Returns the angle whose tangent is the quotient of two specified numbers. | =Atan2(Fields!CoordinateY.Value, Fields!CoordinateX.Value) |

| | | |
|---------|--------------------------------------------------------------------------------------|------------------------------------------------------------|
| BigMul | Produces the full product of two 32-bit numbers. | =`BigMul(Fields!Int32Value.Value, Fields!Int32Value.Value) |
| Ceiling | Returns the smallest integer that is greater than or equal to the specified integer. | =Ceiling(Fields!YearlyIncome.Value / 7) |
| Cos | Returns the cosine of the specified angle. | =Cos(Fields!Angle.Value) |
| Cosh | Returns the hyperbolic cosine of the specified angle. | =Cosh(Fields!Angle.Value) |
| Exp | Returns e raised to the specified power. | =Exp(Fields!IntegerCounter.Value) |
| Fix | Returns an integer portion of a number. | =Fix(Fields!YearlyIncome.Value /-3) |
| Floor | Returns the largest integer less than or equal to the specified integer. | =Floor(Fields!YearlyIncome.Value / 12) |
| Int | Returns an integer portion of a number. | =Int(Fields!YearlyIncome.Value / 12) |
| Log | Returns the natural (base e) logarithm of a specified number. | =Log(Fields!NumberValue.Value) |
| Log10 | Returns the base 10 logarithm of a | =Log10(Fields!NumberValue.Value) |

| | | |
|-------|---------------------------------------------------------------------------------|--------------------------------------------------------------|
| | specified number. | |
| Max | Returns the maximum value from all non-null values of the specified expression. | =Max(Fields!YearlyIncome.Value, "AdventureWorks", Recursive) |
| Min | Returns the minimum value from all non-null values of the specified expression. | =Min(Fields!YearlyIncome.Value, "AdventureWorks", Recursive) |
| Pow | Returns a specified number raised to the specified power. | =Pow(Fields!YearlyIncome.Value, 2) |
| Rnd | Returns a random number of single type. | =Rnd(-1) |
| Round | Rounds a double-precision floating-point value to the nearest integer. | =Round(Fields!YearlyIncome.Value / 12, 2) |
| Sign | Returns a value indicating the sign of an 8-bit signed integer. | =Sign(Fields!YearlyIncome.Value - 60000) |
| Sin | Returns the sine of the specified angle. | =Sin(Fields!Angle.Value) |
| Sinh | Returns the hyperbolic sine of the specified angle. | =Sinh(Fields!Angle.Value) |

| | | |
|------|--------------------------------------------------------|---------------------------|
| Sqrt | Returns the square root of a specified number. | =Sqrt(Fields!Area.Value) |
| Tan | Returns the tangent of the specified angle. | =Tan(Fields!Angle.Value) |
| Tanh | Returns the hyperbolic tangent of the specified angle. | =Tanh(Fields!Angle.Value) |

Inspection Functions

| References | Description | Example |
|------------|----------------------------------------------------------------------------------------|----------------------------------------|
| IsArray | Returns a Boolean value indicating whether variable points to an array. | =IsArray(Parameters!Initials.Value) |
| IsDate | Returns a Boolean value indicating whether an expression represents a valid. | =IsDate(Fields!BirthDate.Value) |
| IsNothing | Returns a Boolean value indicating whether an expression has no object. | =IsNothing(Fields!MiddleInitial.Value) |
| IsNumeric | Returns a Boolean value indicating whether an expression can be evaluated as a number. | =IsNumeric(Fields!YearlyIncome.Value) |

Program Flow Functions

| References | Description | Example |
|------------|-----------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Choose | Selects and returns a value from a list of arguments. | =Tanh(Fields!Angle.Value)=Choose(Datepart("w", Fields!BirthDate.Value), "First", "Second", "Third", "Fourth", "Fifth", "Sixth", "Seventh") |
| IIf | Returns one of two objects depending upon the evaluation of an expression. | =Tanh(Fields!Angle.Value) =IIf(Fields!YearlyIncome.Value >= 60000, "High", "Low") |
| Switch | Evaluates a list of expressions and returns an object value corresponding to the first expression in the list that is true. | =Switch(Fields!FirstName.Value = "Sue", "Susan", Fields!FirstName.Value = "Bob", "Robert") |

Aggregate Functions

| References | Description | Example |
|---------------|---------------------------------------------------------------------------|-------------------------------------------------------------------------|
| Avg | Returns the average of all non-null values from the specified expression. | =Avg(Fields!YearlyIncome.Value, "GroupByGender", Recursive) |
| Count | Returns a count of the values from the specified expression. | =CountDistinct(Fields!MiddleInitial.Value, "GroupByInitial", Recursive) |
| CountDistinct | Returns a count of all distinct values from the specified expression. | =CountDistinct(Fields!MiddleInitial.Value, "GroupByInitial", Recursive) |
| CountRows | Returns a count of rows within the specified scope. | =CountRows("GroupByInitial", Recursive) |
| First | Returns the first value from the specified expression. | =First(Fields!MiddleInitial.Value, "AdventureWorks") |
| Last | Returns the last value from the | =Last(Fields!MiddleInitial.Value, "AdventureWorks") |

| | | |
|--------|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| | specified expression. | |
| Max | Returns the maximum value in the given expression. | =Max(Fields!YearlyIncome.Value, "AdventureWorks", Recursive) |
| Min | Returns the minimum value in the given expression. | =Min(Fields!YearlyIncome.Value, "AdventureWorks", Recursive) |
| StDev | Returns the standard deviation of all non-null values of the specified expression. | =StDev(Fields!YearlyIncome.Value, "GroupByInitial", Recursive) |
| StDevP | Returns the population standard deviation of all non-null values of the specified expression. | =StDevP(Fields!YearlyIncome.Value, "GroupByInitial", Recursive) |
| Sum | Returns a sum of the values of the specified | =Sum(Fields!YearlyIncome.Value, "GroupByInitial", Recursive) |

| | | |
|--------------|------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| | expression. | |
| Var | Returns the variance of all non-null values of the specified expression. | =Var(Fields!YearlyIncome.Value, "GroupByInitial", Recursive) |
| VarP | Returns the population variance of all non-null values of the specified expression. | =VarP(Fields!YearlyIncome.Value, "GroupByInitial") |
| RunningValue | Uses a specified function to return a running aggregate of the specified expression. | =RunningValue(Fields!YearlyIncome.Value, Sum, "AdventureWorks") |
| Aggregate | Returns a custom aggregate of the specified expression, as defined by the data provider. | =Aggregate(Fields!Order_Count.Value) |

Financial Functions

| References | Description | Example |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| DDB | Returns a double value specifying the depreciation of an asset for a specific time period using the double-declining balance method or some other method you specify. | =DDB(Fields!CostOfProperty.Value, Fields!Salvage.Value, Parameters!Life.Value, Parameters!Period.Value, 2) |
| FV | Returns double value specifying the future value of an annuity based on periodic fixed payments and a fixed | =FV(Parameters!Rate.Value, Parameters!NumberOfPayments.Value, Parameters!PaymentAmount.Value, Fields!PropertyCost.Value, DueDate.EndOfPeriod) |

| | | |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| | interest rate. | |
| IPmt | Returns double value specifying the interest payment for a given period of an annuity based on periodic, fixed payments and a fixed interest rate. | =IPmt(Parameters!Rate.Value, Parameters!PaymentPeriod.Value, Parameters!NumberOfPayments.Value, Parameters!PresentValue.Value, 0, DueDate.EndOfPeriod) |
| NPer | Returns a double value specifying the number of periods for an annuity based on periodic fixed payments and a fixed interest rate. | =NPer(Parameters!Rate.Value, Parameters!PaymentAmount.Value, Parameters!PresentValue.Value, 0, DueDate.EndOfPeriod) |
| Pmt | Returns a double | =Pmt(Parameters!Rate.Value, Parameters!NumberOfPayments.Value, Fields!PropertyCost.Value, 0, DueDate.EndOfPeriod) |

| | | |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| | <p>value specifying the payment for an annuity based on periodic, fixed payments and a fixed interest rate.</p> | |
| PPmt | <p>Returns a double value specifying the principal payment for a given period of an annuity based on periodic fixed payments and a fixed interest rate.</p> | <pre>=PPmt(Parameters!Rate.Value, Parameters!Period.Value, Parameters!NumberOfPayments.Value, Fields!PropertyCost.Value, 0, DueDate.EndOfPeriod)</pre> |
| PV | <p>Returns a double value specifying the present value of</p> | <pre>=PV(Parameters!Rate.Value, Parameters!NumberOfPayments.Value, Fields!PaymentAmount.Value, 0, DueDate.EndOfPeriod)</pre> |

| | | |
|------|---------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| | an annuity based on periodic, fixed payments to be paid in the future and a fixed interest rate. | |
| Rate | Returns a double value specifying the interest rate per period for an annuity. | =Rate(Parameters!NumberOfPayments.Value, Parameters!PaymentAmount.Value, Parameters!PresentValue.Value, DueDate.EndOfPeriod, 0.1) |
| SLN | Returns a double value specifying the straight-line depreciation of an asset for a single period. | =SLN(Fields!PropertyCost.Value, Parameters!Salvage.Value, Parameters!Life.Value) |
| SYD | Returns a double value | =SYD(Fields!PropertyCost.Value, Parameters!Salvage.Value, Parameters!Life.Value, Parameters!Period.Value) |

| | |
|-------------------------------------------------------------------------------------|--|
| specifying the sum-of-years digits depreciation of an asset for a specified period. | |
|-------------------------------------------------------------------------------------|--|

Conversion Functions

| References | Description | Example |
|------------|------------------------------------------------------------------|---------------------------------------|
| CBool | Convert to Boolean. | =CBool(Fields!HouseOwnerFlag.Value) |
| CByte | Convert to byte. | =CByte(Fields!Number.Value) |
| CChar | Convert to char. | =CChar(Fields!MaritalStatus.Value) |
| CDate | Convert to date. | =CDate(Fields!BirthDate.Value) |
| CDbl | Convert to double. | =CDbl(Fields!YearlyIncome.Value) |
| CDec | Convert to decimal. | =CDec(Fields!YearlyIncome.Value) |
| CInt | Convert to integer. | =CInt(Fields!YearlyIncome.Value) |
| CLng | Convert to lone. | =CLng(Fields!YearlyIncome.Value) |
| COBJ | Convert to object. | =COBJ(Fields!YearlyIncome.Value) |
| CShort | Convert to short. | =CShort(Fields!NumberCarsOwned.Value) |
| CSng | Convert to single. | =CSng(Fields!YearlyIncome.Value) |
| CStr | Convert to string. | =CStr(Fields!YearlyIncome.Value) |
| Fix | Returns an integer portion of a number. | =Fix(Fields!YearlyIncome.Value / -3) |
| Hex | Returns a string representing the hexadecimal value of a number. | =Hex(Fields!CellColor.Value) |
| Int | Returns an integer portion of a number. | =Int(Fields!YearlyIncome.Value / 12) |
| Oct | Returns a string representing the octal value of a number. | =Oct(Fields!BitString.Value) |
| Str | Returns a string that represents a number. | =Str(Fields!YearlyIncome.Value) |

| | | |
|-----|---------------------------------------------------------------------|---------------------------------|
| Val | Returns numbers in a string as a numeric value of appropriate type. | =Val(Fields!AddressLine1.Value) |
|-----|---------------------------------------------------------------------|---------------------------------|

Miscellaneous Functions

| References | Description | Example |
|------------|---------------------------------------------------------------------|-----------------------------------|
| InScope | Returns true if the current instance is within the specified scope. | =InScope("table1_Group1") |
| Level | Returns a zero-based integer representing the current depth level. | =Level("GroupByInitial") |
| Previous | Returns the value of the expression for the previous row of data. | =Previous(Fields!FirstName.Value) |
| RowNumber | Returns a running count of all rows in the specified scope. | =RowNumber("AdventureWorks") |

Operators in Expressions

An operator is a symbol that represents to perform simple basic operation. The following operators are used in an expression: arithmetic, comparison, concatenation, logical or bitwise, and bit shift.

Arithmetic

Arithmetic operators perform mathematical operations on two numeric terms in an expression.

| Operator | Description |
|----------|----------------------------------------------------|
| $^$ | Raises a number to the power of another number. |
| * | Multiplies two numbers. |
| / | Divides two numbers and returns a float result. |
| | Divides two numbers and returns an integer result. |
| Mod | Returns the integer remainder of a division. |
| + | Adds two numbers. |
| - | Returns the difference between two numbers. |

Comparison

Comparison operators is used to compare whether two expressions are the same.

| Operator | Description |
|----------|------------------------|
| < | Less than. |
| \geq | Less than or equal to. |

| | |
|------|-------------------------------------------------------------------|
| > | Greater than. |
| >= | Greater than or equal to. |
| = | Equal to. |
| <> | Not equal to. |
| Like | Determines specific character string matches a specified pattern. |
| Is | Compares two object references. |

String Concatenation

String concatenation appends the second string to the first string in an expression.

| Operator | Description |
|----------|---------------------------|
| & | Concatenates two strings. |
| + | Concatenates two strings. |

Logical and Bitwise

Logical and bitwise operators perform logical manipulations between two integer terms in an expression.

| Operator | Description |
|----------|-----------------------------------------------------------------------------------------------------------------------|
| And | Performs a logical conjunction on two Boolean expressions, or bitwise conjunction on two. |
| Not | Performs logical negation on a Boolean expression, or bitwise negation on a numeric expression. |
| Or | Performs a logical disjunction on two Boolean expressions, or bitwise disjunction on two numeric values. |
| Xor | Performs a logical exclusion operation on two Boolean expressions, or a bitwise exclusion on two numeric expressions. |
| AndAlso | Performs logical conjunction on two expressions. |
| OrElse | Performs logical disjunction on two expressions. |

Bit Shift

Bitwise operators perform bit manipulations between two integer terms in an expression.

| Operator | Description |
|----------|------------------------------------------------------|
| >> | Performs an arithmetic left-shift on a bit pattern. |
| << | Performs an arithmetic right-shift on a bit pattern. |

Code Module

Code Module is used to add a custom code or custom assemblies to a report. Code module allows to add references to custom code embedded in a report or generated with custom assemblies.

![Open code module](/static/assets/on-premise/images/report-designer/compose-report/code-module/open-code-module.png)

Add Custom Code to a report

1. Click on the **Properties** icon to open **Report Properties**.

![open-report-properties](/static/assets/on-premise/images/report-designer/compose-report/unit-switcher/open-report-properties.png)

2. The custom code option is listed under **Code** category, click on the **Code...** button to open **Code Module** dialog.
3. In the code tab, type the code snippet as shown below. The following code example demonstrates the custom function that return the color string, based on the given value range.

```
'basic
Public Function GetColor(ByVal TotalSales As Integer) As String
Dim ColorName As String
If TotalSales > 0 And TotalSales < 1000 Then
    ColorName = "Yellow"
ElseIf TotalSales > 1000 And TotalSales < 1500 Then
    ColorName = "Blue"
ElseIf TotalSales > 2000 Then
    ColorName = "Red"
Else
    ColorName = "Green"
End If
Return ColorName
End Function
```

![Enter custom code](/static/assets/on-premise/images/report-designer/compose-report/code-module/code-section.png)

4. Click on the **OK** button.
5. Use the following expression, to call the custom code function in the report.

![Call custom code function](/static/assets/on-premise/images/report-designer/compose-report/code-module/custom-code-expression.png)

Refer [Expression](#) section to learn more about handling expressions.

6. To view the color changes based on the price range in a report, click **Preview**.

![Preview custom code](/static/assets/on-premise/images/report-designer/compose-report/code-module/custom-code-preview.png)

Add Assembly References

To embed a custom assemblies in a report, create a simple visual studio class library project and define the required custom functions.

Create a custom assembly

1. Create a visual studio class library. Go to **Installed > Visual C#**, and then select **Class Library (.Net Framework)** from the listed template, change the application name, and then click **OK**.

![To create c sharp class library](/static/assets/on-premise/images/report-designer/compose-report/code-module/assembly-references.png)

2. Open the class file, create a simple function as shown below.

![To create a simple function](/static/assets/on-premise/images/report-designer/compose-report/code-module/add-references.png)

3. After defining the required functions, build the project. Now, the assembly(.dll) will be generated in the bin folder of the respective application.

Note: To embed the custom assembly into the report, the generated assembly must be installed in GAC or it must be available in reporting services **bin** location. We prefer to manually copy and paste the generated custom assembly into the following path **C:\Syncfusion\Report Server\ReportServer.Web\ReportService\bin**.

Adding a references to custom assembly

1. Switch to the **References** tab.

![Code module assembly references](/static/assets/on-premise/images/report-designer/compose-report/code-module/references-tab.png)

2. To add a **Assembly** references, Click on the **Add** icon.

![Add assembly references](/static/assets/on-premise/images/report-designer/compose-report/code-module/reference-add-icon.png)

3. Specify the assembly name and click **OK**.

![Given assembly references name](/static/assets/on-premise/images/report-designer/compose-report/code-module/assembly-reference-name.png)

Add Class instances

1. Switch to the **Classes** tab.

![Switch to class instance tab](/static/assets/on-premise/images/report-designer/compose-report/code-module/classinstance-tab.png)

2. To add a **Class instance** reference, Click on the **Add** icon.

![Code module class instance](/static/assets/on-premise/images/report-designer/compose-report/code-module/listed-class-instance.png)

3. Specify the assembly name, instance name and click **OK**. Now, the custom code assembly and instance are embedded into the report.

![Added class instance name](/static/assets/on-premise/images/report-designer/compose-report/code-module/added-class-instances.png)

4. Use the following expression to call the embedded custom code function in the report.

![Call the method using expression](/static/assets/on-premise/images/report-designer/compose-report/code-module/custom-assembly-expression.png)

Refer [Expression](#) section to learn more about handling expressions.

5. To view the color changes based on the price range in a report, click **Preview**.

![Preview the record](/static/assets/on-premise/images/report-designer/compose-report/code-module/custom-code-preview.png)

Page layout

In web Report Designer, page layout area is categorized as **Header**, **Body**, and **Footer** and it is called as **Design Surface**. You can place report items in the design surface to achieve the desired report design. On report preview and export actions, the report will be generated based on the formatting, page breaks and keep together properties applied to the report items in the design surface.

Rendering layouts

You can export a report to another file formats, such as **Word**, **Excel**, **PDF**, **PowerPoint**, **HTML** or **CSV** using the in-built Bold Report Viewer. Each file format exports reports based on the following three type of layouts.

- Soft page-break layout
- Hard page-break layout
- Data layout

Each layout follows set of predefined rules for report item positioning and determines how much data fits on each page. Thereby, converts data and report items to a readable format.

Soft page-break layout - The report exporting formats such as Word, Excel and HTML follows soft page-break layout. It maintains the report layout and formatting.

To determine how a report will appear in a soft page-break rendering layout, use Preview option. The in-built Bold Report Viewer will display report as it would in a HTML, Word, or Excel format.

Hard page-break layout - The report exporting format such as PDF follows hard page-break layout. It maintains the report layout and formatting.

Reports exported using PDF will have a consistent page size and well suitable for printing purpose.

To determine how a report will appear in a hard page-break rendering layout, use Print preview. The report appears as it would in a PDF format.

Data layout - The report exporting format such as CSV follows data layout. It clears all layout and formatting from the report and display only the data. The exported file can be used to import the raw report data into another file type.

Pagination

Number of pages within a report and how report items are arranged on these pages is termed as pagination. Pagination varies based on the rendering layout used to view and export the report. Reports exported by using the data and soft page page-break layouts are typically not affected by pagination. But, hard page-break have the most impact on report layout and physical page size.

To control pagination, specify the page-related properties which are provided in [Report Properties](#). Each rendering layout supports page properties in different manner, therefore same report will paginate differently depending on the rendering layout used.

- Hard page-break rendering layout is page-oriented format. Therefore, you can set page properties to control the page breaks in reports when viewed in PDF.
- Soft page-break rendering layout is not oriented to physical pages. If the report contains interactive features that causes a report to expand horizontally or vertically, you cannot control the page breaks in reports with Word, Excel and HTML formats. Pagination is applied to reports only in vertical direction. Page margins are not applied.
- Data rendering layout do not support pagination. Page-related properties which are specified in the report will be ignored when you view a report in CSV format.

Paper size

The [paper size](#) that you specify for the report in the Report Properties will define the pagination for the report while rendering and printing report in the print layout.

- Reports exported using hard page break rendering layout inserts page break both horizontally and vertically based on the paper size.

Page break is inserted in horizontal direction when the report body grows outside of the page in right side direction and in vertical direction when the report body grows outside of the page in bottom direction.

- Reports exported using soft page break rendering layout inserts page breaks horizontally based on the paper size.

If specified page size(report width) is lesser than the width of report body, including margins and if rendering layout is Hard page-break the resulting report may flow horizontally across multiple pages.

Margin

Margins are applied from the edge of the physical page dimensions inward to the specified margin setting. The report item will be clipped if it extends into the margin area, so that the overlapping area is not rendered. The margin will be set to zero, if the specified margin sizes causes the horizontal or vertical width of the page to equal zero. Margins are applied only when you render and print reports in hard page break rendering layout.

Avoid the extra blank pages in print and print preview

The extra blank page is created when the width of report is greater the page width. So, to avoid the extra blank pages set the report body width lesser than or equal to the page width.

'html

Body Width <= Page Width - (Left Margin + Right Margin)

The area of the physical page that remains after the space is allocated for margins, column spacing, and page header and footer is called the usable page area. Margins are applied only when you render the report in the print layout and print reports. The following image indicates the margin and usable page area of a physical page.

![Page layout](/static/assets/on-premise/images/report-designer/page-layout.png)

The following formula is used to calculate the usable area of the report rendering:

Horizontal usable area

'html

X = Page.Width - (Left Margin + Right Margin + Column Spacing)

Vertical usable area

'html

Y = Page.Height - (Top Margin + Bottom Margin + Header Height + Footer Height)

For example, consider the report width is 21 cm, the left margin of the report is 0.5 cm, and the right margin of the report is 0.5 cm. To avoid an extra printed page in the exported PDF file, the following formula is used:

'html

width of body (20) + left margin (0.5) + right margin (0.5) <= report width (21)

If the width of body is 20 or lesser, it will be rendered without extra pages. When it uses greater than 20, it will add extra pages.

Report Variables

A report variables option can be used to add custom variables to the report. Its value can be a static text or an expression. You can create report variables when a report design demands a value or calculation to be used many times. Each variable name must be unique in a report.

At run time, the value of report variables is calculated once and is maintained until the report is processed again. You can set a report variable type to be **Read-only** or **Read-write**. By default, it is set to **Read-only** type.

Read-only - Use this type to set constant value for the variable. For example, to create a time stamp. Use the following syntax to call a report variable anywhere in the report, `=Variables!MyVariable.Value`

Read-write - Use this type to assign value dynamically to the variable at run time. When you set the type to read-write, the **Writable** property for that specific variable is set to true. Use the following syntax to set value for a report variable, `=Variables!MyVariable.SetValue("test")`.

Add a report variable

For example, we may want to add report printing time in each page of the report, but using direct expression in a text box will print different time in each page. To get round this problem, we can create a report variable to hold the same time in each page. To create a report variable follow the below steps,

1. The report variables property is listed under Report properties. Click on the Properties icon to open Report Properties.

![Report properties](/static/assets/on-premise/images/report-designer/compose-report/report-variables/report-properties.png)

2. The report variable option is provided under the **Code** category.

![Property category](/static/assets/on-premise/images/report-designer/compose-report/report-variables/variables-option.png)

3. Click on the editor icon, to open the **Variables** panel.

![Open variables panel](/static/assets/on-premise/images/report-designer/compose-report/report-variables/edit-icon.png)

4. Click on the ADD icon in the top right corner, to add new variable field.

![Add new variable field](/static/assets/on-premise/images/report-designer/compose-report/report-variables/add-option.png)

5. Enter the name of the variable in **Name** field.

Note: A name should start with an alphabet character and it should not contain spaces and special characters except underscore.

6. In value field, enter the direct value in the text field or click on the small square icon to enter an expression.
7. Disable the checkbox, to specify the variable type as **Read-write**. By default, it is set to **Read-only**.

Finally, click on the **Update** button in the variable panel to save the report variables state.

![New report variable](/static/assets/on-premise/images/report-designer/compose-report/report-variables/new-variable.png)

You can edit the variable name and value anytime in the report. Open the variables panel and edit the required variables, then click on the update button.

Delete a report variable

To delete the variable from the report, click on the delete icon present on the right side of each variable field and update.

![Delete a report variable](/static/assets/on-premise/images/report-designer/compose-report/report-variables/delete-option.png)

Call report variable in a report

You can call the report variable in any expression. The report variables available in the report are listed under **Data** dropdown in Expression Builder.

![Variables listed in expression builder](/static/assets/on-premise/images/report-designer/compose-report/report-variables/variables-in-expression-builder.png)

Double click on the variable name to insert them into the text editor and click OK.

![Call variable from expression builder](/static/assets/on-premise/images/report-designer/compose-report/report-variables/choose-expression.png)

Configure report items

Bold Report Designer offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

- Basic Items – text box, image, line, and rectangle.
- Data region items – tablix, matrix (pivot), and list.
- Data visualization – 20+ chart types.
- Data bar
- Sparkline
- Indicator
- Linear Gauge
- Radial Gauge
- Subreport.
- Custom report items – barcode and QR code.

See also

[Textbox](#)

[Line](#)

[Image](#)[Rectangle](#)[Tablix](#)[Chart](#)[Data bar](#)[Sparkline](#)[Indicator](#)[Linear Gauge](#)[Radial Gauge](#)[SubReport](#)[Barcode](#)

Line

Line reportitem allows to separate the report sections horizontally, vertically, or diagonally. It has a start and end point with various style properties. A line has visually no data associated with it.

![Line report item in design area](/static/assets/on-premise/images/report-designer/report-items/line/line-reportitem-designarea.png)

To draw a line

1. Drag and drop line reportitem from the itempanel to designer surface.

![Drag and drop line report item in design area](/static/assets/on-premise/images/report-designer/report-items/line/drag-drop-line-reportitem.png)

2. Line report item and its associated properties are listed in the property panel.

![Line report item in design area](/static/assets/on-premise/images/report-designer/report-items/line/line-reportitem-designarea.png)

Line Properties

Line Styles

The line style property can be used to change the style of the line report item. This property is listed under **Basic Settings** category. They are **Solid**, **Dashed** and **Dotted**.

![Line reportitem different styles](/static/assets/on-premise/images/report-designer/report-items/line/line-style-types.png)

Border

The line style property allows to increase or decrease thickness of line report item along three different line styles.

![Line reportitem border styles](/static/assets/on-premise/images/report-designer/report-items/line/line-reportitem-border.png)

Position

The line style property allows the line reportitem to place left and top position in the designer surface.

![Line report item in postion change](/static/assets/on-premise/images/report-designer/report-items/line/line-reportitem-position.png)

Size

The line style property allows to increase or decrease the width and height of line report item.

![Line report item in size change](/static/assets/on-premise/images/report-designer/report-items/line/line-reportitem-size.png)

Visibility

Enable/Disable the visibility property to show and hide the line reportitem, when preview the report.

Miscellaneous

Custom Attributes

This property can be used to set the values for line report item custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Rectangle

The rectangle report item is used to visually separate regions of the report or it can act as container for other report items to improve the overall report design and readability.

Add a rectangle to the report

1. The rectangle report item is listed in the item panel under the **Basic Items** category.

![Rectangle listed in item panel](/static/assets/on-premise/images/report-designer/report-items/rectangle/rectangle-item-in-item-panel.png)

2. Drag and drop the rectangle report item into the design area from the item panel.

![Drag and drop rectangle report item into design area](/static/assets/on-premise/images/report-designer/report-items/rectangle/drag-and-drop-rectangle.png)

3. Once you drop the rectangle item into design area, respective item properties will be listed the properties panel.

![Rectangle item with properties view](/static/assets/on-premise/images/report-designer/report-items/rectangle/rectangle-item-with-properties-view.png)

Properties

Refer the [Properties panel](#) section before proceeding with the below properties.

Basic Settings

The border style, color, width and background color properties are used to style the rectangle and customize its appearance in the report design. These properties are listed under the **Basic Settings** category in the properties panel.

![Rectangle basic settings](/static/assets/on-premise/images/report-designer/report-items/rectangle/rectangle-basic-settings.png)

Border

Border properties are used to add or customize the border around a rectangle item to visually separate items in the report design. To set border properties to the rectangle item using properties panel refer [Border Properties](#) section.

Background color

Using the background color property you can color the rectangle background. To set background color using properties panel refer [Background color](#) section.

Page break

The page break property is used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for rectangle item.

1. Enable the **Enable Page Break** property checkbox.

![Page break property](/static/assets/on-premise/images/report-designer/report-items/rectangle/enable-page-break.png)

2. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

3. To reset page number on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

Position

Position property is used to set the width, height, left and top position of the rectangle in the report design. To handle these properties using properties panel refer [Position](#) section.

Visibility

Visibility property is used to conditionally show or hide the rectangle report item on report preview or export action. To set visibility of rectangle item using properties panel refer [Visibility](#) section.

Miscellaneous

Keep together

Keep together property is used to display the report item or section of a report in a single page, on report preview or export action. Enable the checkbox to keep the content in single page or it will span across multiple pages.

![Keep together property](/static/assets/on-premise/images/report-designer/report-items/rectangle/keep-together-property.png)

Page name

The page name property is used to name the first worksheet of the Excel workbook, when exporting the report to excel format.

| | |
|-------|----------------|
| Image | Set expression |
|-------|----------------|

![Page name property](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-name-property.png)

Miscellaneous

Custom Attributes

This property can be used to set the values for rectangle custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for rectangle report item using properties panel refer [Tooltip](#) section.

Set expression

An expression can be set to few properties of the rectangle report item to process the property values based on expressions. To set expressions to the rectangle report item properties, refer [Set Expression](#) section.

Reset expression

To Reset the expression applied to a property, refer [Reset Expression](#) section.

Advanced properties

Few properties of the rectangle report contains nested properties. To open and handle nested properties, refer [Advanced Properties](#) section.

Image

Image reportitem is used to display images in a report and it can be loaded from external link, data fields and embedded images in report. Images can be of the BMP, GIF, JPG, PNG, or X-PNG type.

Add a image to the report

1. The image reportitem is listed in the item panel under the **Basic Items** category.

![image listed in itempanel](/static/assets/on-premise/images/report-designer/report-items/image/image-reportitem-itempanel.png)

2. Drag and drop the image reportitem into the design area from the item panel.

![Drag and drop image reportitem into design area](/static/assets/on-premise/images/report-designer/report-items/image/image-reportitem-designarea.png)

3. Once you drop the image item into design area, respective item properties will be listed the properties panel.

![image item with properties view](/static/assets/on-premise/images/report-designer/report-items/image/image-properties.png)

Image properties

Basic Settings

Basic Settings contains Source and Value properties. Source includes **Embedded**, **External**, **Database** image type as shown below.

![image reportitem basic setting](/static/assets/on-premise/images/report-designer/report-items/image/basic-properties.png)

- **External**

External image source can be used to display the dynamic image in the report, by specifying a URL to the image or set an [Expression](#) that evaluates to the URL. To add an external image in a report, specify a valid URL in the value property text field.

![image reportitem external type](/static/assets/on-premise/images/report-designer/report-items/image/image-external-type.png)

- **Embedded**

Embedded option is used to display the embedded image that are available in the report. To embed an image in a report, refer [Image Manager](#) section.

![image reportitem basic setting](/static/assets/on-premise/images/report-designer/report-items/image/image-basic-settings.png)

- **Database**

Database option can be used to display images from image data stored in a database. Value and MIME Type properties are listed under Database option.

- **Value**

The available dataset fields in the report will be listed in the value field drop-down. Choose a dataset field that is bound to a database field that contains an image or set an [Expression](#) that evaluates to the image value.

![image reportitem value](/static/assets/on-premise/images/report-designer/report-items/image/image-database-value.png)

- **MIME Type**

MIME property can be used to specify the type of image. Supported MIME types are listed in the MIME property drop-down list. Choose the MIME type as per your image type.

![image reportitem mime type](/static/assets/on-premise/images/report-designer/report-items/image/image-mime-type.png)

Link

Linking is used to create interactive report using **Hyperlink** and **Report Linking** action. Refer [Linking](#) section to know more about linking. Select the **Enable Link** checkbox, to enable the sub properties.

![show-link-action](/static/assets/on-premise/images/report-designer/report-items/image/enable-link-action.png)

Appearance

Appearance is used to improvise the style of an image layout. Border style, color, width properties are listed under the **Appearance** category.

![subreport appearance](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-appearance.png)

Border

Border properties are used to add or customize the border around an image report item to visually separate items in the report design. To set border properties to the image report item using properties panel refer [Border Properties](#) section.

Size

Sizing property allow us to format the size of the images displayed in the design area. It includes four options to adjust the image display, choose the required size in the **Sizing** drop-down list.

- AutoSize
- Fit
- FitProportional
- Clip

![Image reportitem sizing](/static/assets/on-premise/images/report-designer/report-items/image-sizing.png)

AutoSize

AutoSize option will display the original size of an image in the design area.

Fit to Size

Fit option allows to fit the image inside the image reportitem.

Fit Proportional

FitProportional is used to fit the image inside the image reportitem while maintaining certain aspect ratio.

Clip

Clip option is used to display an image from the top left corner of the picture in the image reportitem. If the image is larger than the image reportitem, only a portion of the image is displayed.

Position

Position property is used to set the width, height, left and top position of the image item in the report design. To handle these properties using properties panel refer [Position](#) section.

![subreport item position](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-position.png)

Visibility

Visibility property is used to conditionally show or hide the image report item on report preview or export action. To set visibility of image item using properties panel refer [Visibility](#) section.

Miscellaneous

Custom Attributes

This property can be used to set the values for image report item custom properties. To assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for image report item using properties panel refer [Tooltip](#) section.

Subreport

Subreport is used to embed another report inside the body of a main report. It allows to customize the subreport properties along report path, parameters.

Properties

Basic Settings

Basic Settings category contains Report and Set Parameters properties.

- Report property can be used to specify the report path of the subreport. Refer [Report Path](#) section to set report path to this property.
- Set Parameters property can be used to pass parameters from the main report to the subreport. Refer [Set Parameters](#) section to set parameters to the subreport.

![subreport item basic settings](/static/assets/on-premise/images/report-designer/report-items/subreport/basic-settings.png)

Appearance

Appearance property can be used to improvise the style of subreport layout. Border style, color, width properties are displayed under the Appearance category.

![subreport appearance](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-appearance.png)

Border

Border properties are used to add or customize the border around a subreport item to visually separate items in the report design. To set border properties to the subreport item using properties panel refer [Border Properties](#) section.

No Rows

No Rows property is used to display static text when dataset results with a empty or zero rows.

![subreport item norows property](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-property.png)

Font

Font Styles

To change font style of the no rows message, choose the required font style in the Font Style drop-down list. Following are the supported font styles:

- Default
- Normal
- Italic

![subreport item font style](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-font-style.png)

Font Family

Supported font family names are listed in the drop-down list, choose the required font family from the drop-down list to change the **Font Family** of the no rows message.

![subreport item font family](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-font-family.png)

Font Weight

To change the font weight of the no rows message, choose the required type of font weight property from the drop-down list.

![subreport item font weight](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-font-weight.png)

Font Color

To set font color for the no rows message, click on the color palette icon and choose the required color in the **Color Palette**.

![subreport item font color](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-font-color.png)

Font Size

Font size property allows you to increase the size of the text in the no rows message. You can increase the font size using the numeric drop-down highlighted in below snap.

![subreport item font size](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-font-size.png)

Text Decoration

Text decoration property sets the appearance of decorative lines on text.

- **None** - Produces no text decoration.
- **Underline** - Each line of text is underlined.
- **Overline** - Each line of text has a line above it.
- **Line-through** - Each line of text has a line through the middle.

![subreport item font size](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-text-decoration.png)

Format

Format property is used to format the given text in the no rows message. Refer [Format](#) section to represent the text in **Numbers**, **Currency**, **Date**, **Time**, **Scientific**, **Percentage** and **Custom** formats.

![subreport item font size](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-format.png)

Message

Provide the required text in the message textbox, the given message will be displayed to the user when dataset results with empty or zero rows.

Padding

Padding property is used to provide space around the no rows message content. Increase or decrease the values in numeric dropdown to set the left, right, top and bottom padding to the no rows message.

Line height

Line height property is used to increase space between lines in given message. Increase or decrease the values in numeric dropdown to set the line height for no rows message.

Text Align

The Text align property is used to set the horizontal alignment of a no rows message. The text can be aligned in left, right, or center position. Choose the required text align property from the drop-down list as shown below.

![subreport item text align](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-text-align.png)

Vertical Align

The Vertical Align property is used to set the vertical alignment of a no rows message. Choose the required vertical align property from the drop-down list as shown below.

![subreport item vertical align](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-vertical-align.png)

Writing Mode

Writing mode property represent the direction of the no rows message along Horizontal, Vertical, and Rotate270. Choose the required writing mode property from the drop-down.

Horizontal: Text will be horizontal, read left to right.

Vertical: Text will be vertical, read top to bottom.

Rotate 270: Text will be vertical, read bottom to top.

![subreport item writing mode](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-writing-mode.png)

Position

Position property is used to set the width, height, left and top position of the subreport item in the report design. To handle these properties using properties panel refer [Position](#) section.

![subreport item position](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-norows-position.png)

Visibility

Visibility property is used to conditionally show or hide the subreport report item on report preview or export action. To set visibility of subreport item using properties panel refer [Visibility](#) section.

Miscellaneous

Keep together

Keep together property is used to display the subreport item in a single page, on report preview or export action. Enable the checkbox to keep the content in single page or it will span across multiple pages.

![subreport item keep together](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-keep-together.png)

| | |
|-----------------------------------|----------------|
| Design rdl report using subreport | Set expression |
|-----------------------------------|----------------|

Custom Attributes

This property can be used to set the values for subreport custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for sub-report item using properties panel refer [Tooltip](#) section.

Set expression

Properties panel allows you to specify expressions that can include two or more data fields and various functions. To set expressions for the report or report item properties refer [Set Expression](#) section.

Reset expression

To Reset the property value and expression, refer [Reset Expression](#) section.

Advanced properties

Few properties of the subreport report contains nested properties. To open and handle nested properties, refer [Advanced Properties](#) section.

Design rdl report using subreport

Refer [Design rdl report using subreport](#) section to learn how to link report, set parameters, to customize the subreport properties.

Design rdl report using subreport

This section describes the steps to create SSRS RDL report to compare two different employees details side by side using subreport.

Create a main report

1. Below is the dataset query used for the main report. Refer [Create Data](#) section and create dataset using the below query.

```
'sql
SELECT
Employees.EmployeeID,
Employees.FirstName
FROM
Employees
` 

![create a dataset](/static/assets/on-premise/images/report-designer/report-items/subreport/main-report-dataset.png)
```

2. To compare the employee details, create two parameters named Employee1 and Employee2 in the main report and assign the required dataset field value to the parameters as shown in the

following snaps. Refer [Parameter](#) and [Assign Value](#) section to create and assign value to the parameters.

![Create a parameter](/static/assets/on-premise/images/report-designer/report-items/subreport/main-report-parameter.png)

![Assign parameter values](/static/assets/on-premise/images/report-designer/report-items/subreport/assign-parameter-values.png)

3. Using textbox and line report items, design a simple report as shown below.

![create a main report](/static/assets/on-premise/images/report-designer/report-items/subreport/main-report-textbox.png)

Note: Refer [Textbox](#), [Line](#) section to design the above report design.

4. Drag and drop subreport item listed in the item panel under the **Sub Reports** category.

![Subreport listed in item panel](/static/assets/on-premise/images/report-designer/report-items/subreport/subreportitem-itempanel.png)

5. To compare the employee details, drag and drop two subreport items side by side in the design area.

![Drag and drop subreportitem into design area](/static/assets/on-premise/images/report-designer/report-items/subreport/subreportitem-designarea.png)

[Create a subreport](#)

Design a subreport to display the employee details.

[Create dataset](#)

1. Below is the dataset query used in this report to fetch the employee personal details that is bound to the **Employees** table of **Northwind** database. Refer [Create Data](#) section and create dataset using the below query.

```
'sql
SELECT
Employees.EmployeeID,Employees.LastName,Employees.FirstName,Employees.Title,Employees.TitleOfC
ourtesy ,(DATENAME(WEEKDAY,Employees.BirthDate)+', '+ DATENAME (DAY,Employees.BirthDate)+'
'+DATENAME(MONTH,Employees.BirthDate)+' '+DATENAME(YEAR,Employees.BirthDate)) as BirthDate
,(DATENAME(WEEKDAY,Employees.HireDate)+', '+DATENAME(DAY,Employees.HireDate)+'
'+DATENAME(MONTH,Employees.HireDate)+' '+DATENAME(YEAR,Employees.HireDate))as HireDate
,Employees.HomePhone ,Employees.City,Employees.Region,Employees.PostalCode,Employees.Country
,SUM(o.Quantity * o.UnitPrice) As TotalGain FROM
Employees,[Order Details] as O,[Orders] as Ord WHERE o.OrderID = ord.OrderID and
Employees.EmployeeID = ord.EmployeeID and Employees.EmployeeID = @SalesPersonID
```

```
group by Employees.EmployeeID
,Employees.LastName,Employees.FirstName,Employees.Title,Employees.TitleOfCourtesy,BirthDate,Hire
Date,Employees.City,Employees.HomePhone,Employees.Region
,Employees.PostalCode,Employees.Country
```

2. To present respective employee sales order details information in the tabular format, create a dataset using below query and bind data to the table report item.

`sql

```
SELECT Top(10) o.OrderID,Cus.CompanyName,SUM(OD.UnitPrice * od.Quantity ) As ExPrice FROM
[Orders] as O, [Customers] as Cus,[Order Details] as OD where (O.EmployeeID = @SalesPersonID) and
(cus.CustomerID=o.CustomerID) and od.OrderID = o.OrderID group by
o.OrderID,o.CustomerID,cus.CompanyName
```

3. Use the below query and create a dataset, to retrieve the employee image details from Employees table of Northwind database.

`sql

```
SELECT Employees.Photo FROM Employees
WHERE (Employees.EmployeeID = @SalesPersonID)
```

Now, the dataset created in the report will be listed in the **Data panel** as shown below.

![Subreport three dataset](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-three-dataset.png)

4. Based on the **@SalesPersonID** query parameter created in dataset query, equivalent report parameter will be created automatically in the report as shown below.

![Subreport parameter](/static/assets/on-premise/images/report-designer/report-items/subreport/subreport-parameter.png)

Design report

Here we have designed a simple subreport using image, textbox and table reportitem as shown below.

![subreport item with properties view](/static/assets/on-premise/images/report-designer/report-items/subreport/create-a-subreport.png)

Note: Refer [Image](#), [Line](#), [Table](#) section to design the above subreport.

Assign fields

- To display employee DOB, we have used this
`=First(Fields!BirthDate.Value,"EmployeePersonalDetails")` expression in the textbox reportitem.
- To display employee DOJ, we have used this
`=First(Fields!HireDate.Value,"EmployeePersonalDetails")` expression in the textbox reportitem.
- To display employee GAIN we have used this
`=Sum(Fields!TotalGain.Value,"EmployeePersonalDetails")` expression in the textbox reportitem.
- To display employee First Name and Last Name, we have used this
`=First(Fields!FirstName.Value, "EmployeePersonalDetails") + " " +`
`" + First(Fields!LastName.Value, "EmployeePersonalDetails")` expression in the textbox reportitem.
- To present the sales details of a respective employee, EmployeeSalesDetails dataset is assigned to the table report item.
- To display employee image in the report, assigned the `=First(Fields!Photo.Value,`
`"EmployeeImages")` expression to the image report item value property.

![subreport item with properties view](/static/assets/on-premise/images/report-designer/report-items/subreport/assign-value-to-image-report-item.png)

Link sub report into main report

Once you drag and drop the subreport item into design area, respective item properties will be listed in the properties panel.

![subreport item with properties view](/static/assets/on-premise/images/report-designer/report-items/subreport/subreportitem-properties.png)

Follow the below steps to link subreport into main report

1. Click on the browse button in **Report** option as shown below.

![Report link-fields](/static/assets/on-premise/images/report-designer/report-items/subreport/report-linking-option.png)

2. Browse dialog will open, click on **Sample Reports** folder.

![browse-from-dialog](/static/assets/on-premise/images/report-designer/report-items/subreport/sample-folder.png)

3. Select the already designed sub report and click **Open**.

![select report from folder](/static/assets/on-premise/images/report-designer/report-items/subreport/browse-report-dialog.png)

4. Selected report path is linked in the **Report** option and subreport item in the design area as shown below.

![selected report path](/static/assets/on-premise/images/report-designer/report-items/subreport/report-path.png)

Similarly set the report path to the another subreport item by following the above steps.

Set Parameters values

To assign parameter values to the sub report from main report follow the below steps:

1. Click on the **Set Parameters** button and click on the **ADD** icon.

![selected report path](/static/assets/on-premise/images/report-designer/report-items/subreport/parameter-add-icon.png)

2. The available parameters in the sub report will be listed in the **Parameters Name** drop down, choose the parameter name and assign values. Here, the following expression `=Parameters!Employee1.Value` is assigned to the **SalesPersonID** parameter of subreport.

![Pass the parameter value](/static/assets/on-premise/images/report-designer/report-items/subreport/parameter-values-passed.png)

3. Similarly set the parameter name and assign values to the another subreport item by following the above steps.

Now, the report path and parameters are set for both subreport item in the report as shown below snap.

![side by side subreport](/static/assets/on-premise/images/report-designer/report-items/subreport/side-by-side-subreport.png)

Preview the report and choose the required employee name in the **Left Side Employee** and **Right Side Employee** parameter drop-down and click on view report. On report preview, the comparison report for two employees will be shown based on the values selected in the parameter drop-down.

![Preview the highlighted parameter](/static/assets/on-premise/images/report-designer/report-items/subreport/preview-highlighted-parameter.png)

TextBox

TextBox can be used to display static text for titles, highlighting key information, descriptions and labels or dynamic text set based on expressions.

Add a textbox to the report

1. The textbox report item is listed in the item panel under the **Basic Items** category.

![Textbox listed in item panel](/static/assets/on-premise/images/report-designer/report-items/textbox/textbox-in-item-panel.png)

2. Drag and drop the textbox report item into the design area from the item panel.

![Drag and drop textbox report item into design area](/static/assets/on-premise/images/report-designer/report-items/textbox/drag-and-drop-textbox.png)

3. Once you drop the textbox item into design area, respective textbox properties will be listed in the properties panel.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/textbox-item-with-properties-view.png)

Properties

In textbox report item you can view and set properties for the textbox report item and each content in the textbox. (i.e Paragraph).

To view and edit overall textbox properties select the textbox report item in design area. Now, the textbox properties will be listed in the properties panel like below.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/textbox-properties.png)

To view and edit the selected text properties, focus inside the text area. Now, the selected text properties will be listed in the properties panel like below.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/selected-text-properties.png)

Refer the [Properties panel](#) section before proceeding with the below properties.

Common properties

The following are the common properties for textbox and selected text:

Basic Settings

Font

The font family, font size, font color, font style, and font weight properties are used to style the content in the textbox. You can apply these font properties to the over all textbox content or specific content in a textbox.

Font Family

Supported font family names are listed in the drop-down under **Basic Settings** category. To change the **Font Family** of the textbox, choose the required font family from the drop-down list.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/font-family-drop-down.png)

Font Color

To set font color for the content in the textbox, click on the color palette icon and choose the required color in the **Color Palette**.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/font-color-picker.png)

Font Size

Font size property allows you to increase the size of the text in the textbox. You can increase the font size using the numeric drop-down highlighted in below snap.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/font-size-numeric-drop-down.png)

Font Styles

To change font style of the content in textbox, choose the required font style in the **Font Style** drop-down list. Following are the supported font styles:

- Default
- Normal
- Italic

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/font-style-drop-down.png)

Font Weight

To change the font weight of the content in textbox, choose the required type of font weight property from the drop-down list.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/font-weight-property.png)

Text Decoration

Text decoration property sets the appearance of decorative lines on text.

- **None** - Produces no text decoration.
- **Underline** - Each line of text is underlined.
- **Overline** - Each line of text has a line above it.
- **Line-through** - Each line of text has a line through the middle.

Underline and Overline decorations are positioned under the text, line-through over it.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/text-decoration.png)

Format

Format property is used format an entire text box or specific text, numbers, expressions, or fields within the text box. To open the format dialog click on the icon highlighted in the below snap or you can type the format directly in the textbox.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/format-button.png)

To apply **Number, Currency, Date, Time, Scientific, Percentage** or **Custom** formats using format dialog follow the steps provided in [Format](#) section.

[Link](#)

You can specify a hyperlink inside the textbox content to link another report. Select the **Enable Link** checkbox, to enable the sub properties.

![Border width](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking.png)

To set the hyperlink to a selected text or over all textbox follow the steps provided in [Linking](#) section.

Textbox properties

The following properties are only specific to the over all textbox item:

Alignment

Text Alignment

You can align the textbox content or specific content in a textbox using the **Text Alignment** property. Choose the respective text alignment property from the drop-down list to align the text in **Left**, **Right**, or **Center** positions.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/text-alignment-property.png)

Vertical alignment

You can align the textbox content or specific content in a textbox in **Top**, **Middle**, and **Bottom** positions. Choose the required vertical alignment property from the drop-down list.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/vertical-alignment.png)

Line height

Line height property is used to increase space between lines of a textbox. Increase or decrease the values in numeric dropdown to set the line height.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/line-height-property.png)

Appearance

The border style, color, width and background color properties are used to style the textbox and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Textbox appearance settings](/static/assets/on-premise/images/report-designer/report-items/textbox/textbox-appearance.png)

Background color

Using the background color property you can color the textbox background. To set background color using properties panel refer [Background color](#) section.

Border

Border properties are used to add or customize the border around a textbox item to visually separate items in the report design. To set border properties to the textbox item using properties panel refer [Border Properties](#) section.

Position

Position property is used to set the width, height, left and top position of the textbox in the report design. To handle these properties using properties panel refer [Position](#) section.

Visibility

Visibility property is used to conditionally show or hide the textbox report item on report preview or export action. To set visibility of textbox item using properties panel refer [Visibility](#) section.

Miscellaneous

![Visibility property](/static/assets/on-premise/images/report-designer/report-items/textbox/miscellenous-property.png)

Can Grow

Enable this property to expand the Textbox height vertically based on their content while previewing the text.

Can Shrink

Enable this property to shrink the Textbox height vertically based on their content while previewing the text.

Custom Attributes

This property can be used to set the values for textbox custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for textbox report item using properties panel refer [Tooltip](#) section.

Localization

Direction

The **Direction** property can be used to configure the content rendering direction for textbox report item. On preview, content in the textbox will be rendered based on the provided direction. You can set either Right-To-Left or Left-To-Right direction.

![Text direction](/static/assets/on-premise/images/report-designer/report-items/textbox/direction.png)

By default, content renders in Left-To-Right direction.

Left-To-Right report preview:

![Left-To-Right direction](/static/assets/on-premise/images/report-designer/report-items/textbox/left-to-right.png)

Right-To-Left report preview:

![Right-To-Left direction](/static/assets/on-premise/images/report-designer/report-items/textbox/right-to-left.png)

Language

The **Language** property can be used to set the locale on a textbox which determines the default formats for displaying report data in textbox. Select the required language in the **Language** property dropdown.

![Report Language](/static/assets/on-premise/images/report-designer/report-items/textbox/language.png)

The language property on a text box overrides the language property on the report.

Selected Text properties

Markup Type

Markup type property can be used to render the selected text as plain text or HTML-formatted text.

Select the required text and choose the type in **Markup Type** property dropdown.

![Textbox Markup Type](/static/assets/on-premise/images/report-designer/report-items/textbox/markup-type.png)

Plain Text - Displays the selected text as simple text and the HTML will be treated as literal text.

HTML - Displays the selected text as HTML. If the expression value or values from dataset field contains a valid HTML tags, these tags will be rendered as HTML. Refer [Add HTML in to a report](#) section to import HTML using textbox.

Paragraph Settings

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/paragraph-settings.png)

Text Alignment

You can align the textbox content or specific content in a textbox using the **Text Alignment** property. Choose the required text alignment property from the drop-down list.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/paragraph-settings-text-alignment.png)

Indent

To indent the text in the paragraph you can use this property. Increase/decrease the left and right indent in the text box using the numeric drop-down.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/indent-property.png)

Space

Using this property you can add space before and after the paragraphs in the textbox.

![Textbox item with properties view](/static/assets/on-premise/images/report-designer/report-items/textbox/space-property.png)

Set expression

An expression can be set to few properties of the textbox report item to process the property values based on expressions. To set expressions to the textbox report item properties, refer [Set Expression](#) section.

Reset expression

To **Reset** the expression applied to a property, refer [Reset Expression](#) section.

Advanced properties

Few properties of the textbox report contains nested properties. To open and handle nested properties, refer [Advanced Properties](#) section.

Design RDL report using textbox

Refer [Design RDL report using textbox](#) section to learn how to position, style, format, link report, add hyperlink in a textbox.

Design RDL report using textbox

This section describes the steps to design RDL report using textbox.

Add textbox to the report

You can add textbox report item in header, footer and body area of the report.

1. Refer [Add a textbox to the report](#) section and add a textbox in the report header, body and footer section.
2. In the below snap, the textbox is added in report header to display a title text, report body to display static fields, and in report footer to display a label for signature field in the report.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/enter-text-in-footer-textbox.png)

Display dynamic text using expression

You can display dynamic values in the textbox using parameters and dataset fields. To achieve this requirement expressions are used. In the below steps, the report parameter is assigned in the textbox to display the dynamic value.

1. Refer [Create Parameter](#) section and create a new report parameter
2. Refer [Add a textbox to the report](#) section and add a textbox in the report.
3. Focus in the textbox and right click in the text area. The textbox context menu will appear as shown below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/expression-menu.png)

4. Select the **Expression** option in the menu list. Now, expression editor opens as shown below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/expression-editor.png)

5. You can choose/specify the required expression in the text area. Here, a **InvoiceID** parameter appended with **#** symbol is assigned as expression.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-parameter-value-as-expression.png)

To learn more about handling expressions in report designer refer [Expressions](#) section.

6. The specified expression is displayed as **<>** in the text area at respective cursor position.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/expression-created.png)

Edit the expression

1. To edit the created expression in the textbox, select the specific **<>** tag.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/select-expression-tag.png)

2. Right click on the respective tag to open the context menu.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/open-textbox-context-menu.png)

3. Select the **Expression** option in the menu list. Now, expression editor opens as shown below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-parameter-value-as-expression.png)

4. Now, modify the expression as required and click on the **OK** button.

Associate a textbox with dataset

Refer [Create Data](#) section to create dataset in Web Report Designer.

Drag and drop dataset field

1. Open the **DATA** panel and expand the required dataset fields

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/dataset-list-view.png)

2. Drag and drop the required dataset field into the design area,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/drag-and-drop-dataset-field.png)

3. Now, the new textbox will be added in the design area like below,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/drag-and-drop-dataset-field-output.png)

Assign the field using expression editor

You can also associate the dataset field in textbox using the expression editor. Refer [Dataset Fields in Expression](#) section.

Associate a textbox with parameter

Refer [Create Parameter](#) section to create report parameter in Web Report Designer.

Drag and drop parameter field

1. Open the **PARAMETERS** panel, the available parameters in the report will be listed in the panel.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/parameter-list-view.png)

2. Drag and drop the required parameter into the design area,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/drag-and-drop-parameter.png)

3. Now, the new textbox will be added in the design area like below,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/drag-and-drop-parameter-output.png)

Assign the parameter using expression editor

You can also associate parameter in textbox using the expression editor. Refer [Dataset Fields in Expression](#) section.

Configure and Format textbox content

You can configure the textbox in below ways:

- Focus the overall textbox element to configure properties for whole textbox content.
- Select the specific text to configure properties for selected text.

Position and Sizing

To improve the report readability and design, you can set position and sizing properties for the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/position-and-sizing.png)

Report design view after sizing the textboxes,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/position-and-sizing-output.png)

Style textbox content

Over all textbox

Select the textbox and open the properties panel. Under the **Basic Settings** and **Appearance** category set the font and alignment properties for the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/style-over-all-textbox.png)

Selected text

Select specific text in the textbox and open the properties panel. Now, the **Selected Text** properties will be displayed in the properties panel.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/style-selected-text.png)

Under the **Basic Settings** and **Paragraph Settings** category set the font and alignment properties for the selected text.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/style-selected-text-output.png)

Format value in textbox

You can apply format for overall textbox value or to a selected text in a textbox.

Format overall textbox

1. Select a textbox in the design area. In the below image, the textbox contains **UnitPrice** dataset field is selected.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/format-dataset-field.png)

2. In the properties pane, to set the format click on the below highlighted button under the **Basic Settings** category.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/format-dataset-field-button.png)

3. Now, the format dialog will open. Refer [Format Data](#) section to set different formats to the data. Here, the **Currency** format is applied to the **Unit Price** dataset field.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-format-to-overall-textbox.png)

4. Once you set format, the format will be shown in the properties panel like below,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/format-value-in-text-area.png)

5. On report preview, the **Unit Price** field value will be displayed like below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/format-report-preview.png)

Format selected text

1. Select a specific text in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/apply-format-for-selected-text.png)

2. In the properties pane, to set the format click on the below highlighted button under the **Basic Settings** category.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/selected-text-format-button.png)

3. Now, the format dialog will open. Refer [Format Data](#) section to set different formats to the data. Here, the **Currency** format is applied to the **Unit Price** dataset field.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-format-to-overall-textbox.png)

4. Once you set format, the format will be shown in the properties panel like below,

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/format-value-in-text-area.png)

5. On report preview, the **Unit Price** field value will be displayed like below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/selected-text-format-preview.png)

Report linking

You can link a report in the textbox, when you click on a specific text it will display the respective report in the report viewer.

Over all textbox

1. Select a textbox in the design area.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/select-a-textbox-to-link-report.png)

2. In the properties view, enable the **Enable Link** option under **Link** category.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/enable-link-over-all-textbox.png)

3. Refer [Report Linking](#) section and set the report path in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-report-path.png)

4. On report preview, when you click anywhere within the specific textbox will display the **SalesOrderDetail** report in the report viewer.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-overall-textbox-preview.png)

5. Once you click on the textbox, the target report will be displayed like below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-overall-textbox-output.png)

Selected text

1. Select a specific text in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-selected-text.png)

2. In the properties view, enable the **Enable Link** option under **Link** category.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-selected-text-enable-link.png)

3. Refer [Report Linking](#) section and set the report path in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-report-path.png)

4. On report preview, when you click on the **here** text in the respective textbox will display the [SalesOrderDetail](#) report in the report viewer.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-selected-text-preview.png)

5. Once you click on the textbox, the target report will be displayed like below.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-overall-textbox-output.png)

Add hyperlink

You can define the events to take place when users click on the text box or certain area displayed on a report.

Over all textbox

1. Select a textbox in the design area.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/add-hyper-link-select-textbox.png)

2. In the properties view, enable the **Enable Link** option under **Link** category.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/enable-link-over-all-textbox.png)

3. Refer [Hyperlink](#) section and set the URL in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-report-url.png)

4. On report preview, when you click anywhere within the specific textbox will display the **SalesOrderDetail** report in the report viewer on the new tab in browser.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/hyperlink-report-preview.png)

Selected text

1. Select a specific text in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-selected-text.png)

2. In the properties view, enable the **Enable Link** option under **Link** category.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/report-linking-selected-text-enable-link.png)

3. Refer [Hyperlink](#) section and set the URL in the textbox.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/set-report-url.png)

4. On report preview, when you click on the **here** text in the respective textbox will display the **SalesOrderDetail** report in the report viewer on the new tab in browser.

![Design report using textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/hyperlink-report-preview.png)

Add HTML in to a report

The text box report item allows you to insert HTML-formatted text into a report. The text can be any simple or complex expression, or a value retrieved from a field in your dataset.

Format plain text as HTML

Drag and drop a textbox report item in to design surface.

![Add textbox](/static/assets/on-premise/images/report-designer/report-items/textbox/html-format-text/add-textbox.png)

Enter the text in valid HTML format into text area.

![Enter valid html text](/static/assets/on-premise/images/report-designer/report-items/textbox/html-format-text/set-html-text.png)

Select the text and set **HTML** in **Markup Type** property.

![Set Markup Type](/static/assets/on-premise/images/report-designer/report-items/textbox/html-format-text/set-markup-type.png)

You can also format a dataset field which returns HTML text from database or an expression which evaluates to a text in HTML format using the markup property.

On report preview, the text will be rendered as shown below,

![Markup type preview](/static/assets/on-premise/images/report-designer/report-items/textbox/html-format-text/preview-html-text.png)

Invalid HTML markup tags and cascading style sheet attributes will be ignored and render as plain text. Following are the few set of HTML tags and cascading style sheet attributes which will render as HTML when defined with HTML markup type,

```
<ul>
<li><p>Hyperlinks: <A HREF></p></li>
<li><p>Fonts: <FONT></p></li>
<li><p>Header, style and block elements: <H{n}>, <DIV>, <SPAN>, <P>, <DIV>, <LI>, <HN></p></li>
<li><p>Text format: <B>, <I>, <U>, <S></p></li>
<li><p>List handling: <OL>, <UL>, <LI></p></li>
<li><p>text-align, text-indent</p></li>
<li><p>font-family</p></li>
<li><p>font-size</p></li>
<li><p>color</p></li>
<li><p>padding, padding-bottom, padding-top, padding-right, padding-left</p></li>
<li><p>font-weight</p></li>
</ul>
```

Tablix

Tablix report item is used to display paginated report data from a dataset in cells that are organized into rows and columns. It is the combination of Table, Matrix, and List report items (Table + Matrix + List = Tablix). The Tablix report items are listed under Data Regions category in the item panel.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/item-panel-initial-view.png)

Matrix report item is not provided as separate report item. The table report item can be used to create a matrix layout.

The following section explains about these report items.

Table

Table can be used to display data in tabular format. A simple table design contains a table header row, and a details row with three columns. Table can have only row groups.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/table-initial-design.png)

Matrix

Matrix can be used to display summarized data. It can have row groups and column groups. A simple matrix design contains a row group, a column group, a corner cell, and a data cell.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/initial-matrix-design.png)

List

List report item can be used to create free-form layouts. It acts as a container to place multiple report items side by side to design a free-form layout.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/initial-list-design.png)

Tablix sections

Tablix data region can be classified into four sections:

- Corner cell - It is created when tablix data region has both row groups and column groups.
- Row group cells - Displays group instance values
- Column group cells - Displays group instance values
- Body cells - Displays detail and group data

The following snap shows the areas for a tablix region with nested row groups and column groups

![Tablix areas](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-areas.png)

The tablix body area always exists in the tablix data region. The other areas are optional.

Groups and total

When you select a tablix cell, row and column grippers and group indicators inside the tablix data region will show the groups to which the respective cell belongs.

The following snap shows a matrix with both row and column groups, and a total row and a total column.

![Groups and total](/static/assets/on-premise/images/report-designer/report-items/tablix/groups-and-total-sketch.png)

Properties

Refer the [Properties panel](#) section before proceeding with the below properties.

Data

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/data-category-property.png)

Dataset

This property is used to assign the dataset to the tablix. The available datasets in the report will be listed in the **Dataset** property dropdown. You can choose the desired dataset from the drop-down.

Each tablix report item can only show data from one dataset.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/dataset-drop-down-view.png)

Refer [Create Data](#) section to add dataset to your report.

Filter

Filters is used to filter the data in the tablix. To open the **Filter** dialog, click on the **Set Filters...** button. Now, the filter dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/filters-dialog.png)

Refer [Filter Data](#) section to add/remove filters in the filter dialog.

Sort

To sort the numeric or string field in the tablix, sorting can be used. In tablix, the sorting can be applied to the whole data region or for each group, including the details group. To open the sort dialog, click on the **Set Sorts...** button. Now, the sort dialog will be opened like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/sort-dialog.png)

Refer [Sort Data](#) section to add/remove sort expressions in the sort dialog.

Appearance

The border style, color, width and background color properties are used to style the tablix and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

Border

Border properties are used to add or customize the border around a tablix item to visually separate it in the report design. To set border properties to the tablix item using properties panel refer [Border Properties](#) section.

Background color

Using the background color property you can color the tablix background. To set background color using properties panel refer [Background color](#) section.

Page break

Page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for tablix report item.

![Tablix keep together](/static/assets/on-premise/images/report-designer/report-items/tablix/keep-together.png)

Break location

The Break Location property specifies where the page break should occur. Choose the required **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

Page number reset

To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

Keep together

Enable this checkbox, to keep the entire tablix together on one page if possible.

![Keep together tablix](/static/assets/on-premise/images/report-designer/report-items/tablix/keep-together.png)

Headers

Headers property can used to configure the tablix row and column headers behaviour when previewing the report.

![Tablix headers](/static/assets/on-premise/images/report-designer/report-items/tablix/headers.png)

Fixed Row

Fixed row property can be used to freeze the row headers while scrolling the pages of a report.

Fixed Column

Fixed column property can be used to freeze the column headers while scrolling the pages of a report.

Repeat Row

Repeat row property can be used to show the row header on every single page of the report.

Repeat Column

Repeat column property can be used to show the column header on every single page of the report.

Position

Position property is used to set the width, height, left and top position of the tablix in the report design. To handle these properties using properties panel refer [Position](#) section.

Visibility

Visibility property is used to conditionally show or hide the tablix report item on report preview or export action. To set visibility of tablix item using properties panel refer [Visibility](#) section.

Miscellaneous

Custom Attributes

This property can be used to set the values for tablix custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for table report item using properties panel refer [Tooltip](#) section.

Set expression

An expression can be set to few properties of the tablix report item to process the property values based on expressions. To set expressions to the tablix report item properties, refer [Set Expression](#) section.

Reset expression

To Reset the expression applied to a property, refer [Reset Expression](#) section.

Advanced properties

Few properties of the tablix report items contains nested properties. To open and handle nested properties, refer [Advanced Properties](#) section.

Design RDL report using table

Refer [Design ssrs RDL report using tablix](#) section to learn how to design a simple tablix in your report.

Grouping Panel

The grouping panel displays the row and column groups of the selected tablix data region in design area. The highlighted area in the below snap shows the grouping panel in design area.

![Grouping panel](/static/assets/on-premise/images/report-designer/report-items/tablix/grouping-panel.png)

The row groups of a selected tablix are listed in the **Row Groups** section and column groups are listed in the **Column Groups** section of the grouping panel.

Show or hide grouping panel

You can expand or collapse the grouping panel in the design area at design time using the expand/collapse icon provided in the right corner of the grouping panel header.

![Expand collapse icon](/static/assets/on-premise/images/report-designer/report-items/tablix/grouping-panel-expand-collapse-icon.png)

The below snap shows the collapsed view of grouping panel in the design area.

![Grouping panel collapse state](/static/assets/on-premise/images/report-designer/report-items/tablix/grouping-panel-collapsed-state.png)

Grouping panel will be visible in design area only if a tablix report item is selected in the design area.

Resize grouping panel

You can increase or decrease the height of the grouping panel in the design area using the resizer.

![Grouping panel resizer](/static/assets/on-premise/images/report-designer/report-items/tablix/grouping-panel-resizer.png)

Modes of grouping panel

Grouping panel have **Default** and **Advanced** mode.

Default Mode

In default mode, the grouping panel displays the hierarchical view of parent, child and adjacent groups of selected tablix item in **Row Groups** and **Column Groups** pane.

![Group types sketch](/static/assets/on-premise/images/report-designer/report-items/tablix/group-types-sketch.png)

Advanced Mode

By default, grouping panel will be displayed in **Default Mode**, to enable advanced mode in the grouping panel, click on the icon in the right corner of the grouping panel header.

![Advanced mode menu icon](/static/assets/on-premise/images/report-designer/report-items/tablix/advanced-mode-menu-icon.png)

Then, click on the Advanced menu item. Now, all groups including static and dynamic tablix members will be listed in Row Groups and Column Groups pane in hierarchical view like below.

![Advanced mode members view](/static/assets/on-premise/images/report-designer/report-items/tablix/grouping-panel-with-static-and-dynamic-tablix-members-view.png)

Once you enable the advanced mode, a tick mark will be indicated in the Advanced menu like below.

![Advanced mode menu icon](/static/assets/on-premise/images/report-designer/report-items/tablix/advanced-mode-indication.png)

Visual Cues

Groups

The row and column groups in the selected tablix data region will be indicated in the grouping panel using the [symbol.

![Group indication](/static/assets/on-premise/images/report-designer/report-items/tablix/visual-cues-group-indicator.png)

Details Group

Details group on the row group hierarchy will be indicated in the grouping panel using the highlighted symbol in the below snap.

![Details Group Indication](/static/assets/on-premise/images/report-designer/report-items/tablix/visual-cues-detail-group-indicator.png)

Label convention

In the grouping panel,

Static tablix member

- A static member with a header cell will be indicated with Static label text.

![Static member with header cell](/static/assets/on-premise/images/report-designer/report-items/tablix/static-member-with-header-cell.png)

- A static member with no header cell will be indicated with (Static) label text.

![Static member with header cell](/static/assets/on-premise/images/report-designer/report-items/tablix/static-member-with-no-header-cell.png)

Group tablix member

The label text for group member will be surrounded by square brackets.

![Static member with header cell](/static/assets/on-premise/images/report-designer/report-items/tablix/grouping-panel-group-member-lable-indication.png)

Tablix member properties

To edit the static and dynamic tablix member properties, select the respective member in the Row or Column Groups pane, and change the property values in the Properties panel.

Static member properties

Click on the required static tablix member in the Row Groups or Column Groups pane, now the respective tablix member properties will be listed in the properties panel.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/open-static-member-properties.png)

To set and edit the properties of static tablix member, refer [Static member properties](#) section.

Group member properties

Click on the required group tablix member in the Row Groups or Column Groups pane, now the respective tablix member properties will be listed in the properties panel.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/open-group-member-properties.png)

To set and edit the properties of group tablix member, refer [Group member properties](#) section.

Add groups or total

You can add groups or totals to the tablix data region using the grouping panel. To open the group menu click on the icon in the right corner of the row or column group member.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/group-menu-icon.png)

Now, the group menu will open in the respective group like below,

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/group-menu-open-view.png)

Add parent group

1. To add a parent group, click on the Parent Group... option under Add Group category.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/parent-group-menu.png)

2. Now, Tablix Group dialog will open like below.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-group-dialog.png)

3. Choose field in the group dialog and click OK button.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/choose-field-in-group-dialog.png)

4. Now, a new parent group will be added above a group in the grouping panel.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/new-parent-group.png)

Add child group

1. To add a child group, click on the **Child Group...** option under **Add Group** category.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/child-group-menu.png)

2. Now, **Tablix Group** dialog will open like below.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-group-dialog.png)

3. Choose field in the group dialog and click **OK** button.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/choose-field-in-group-dialog.png)

4. Now, a new child group will be added under a group in the grouping panel.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/child-group-output.png)

You cannot add child group for **Details** group.

Add adjacent group

1. To add a adjacent group, click on the **Adjacent Before** or **Adjacent After** option under **Add Group** category.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/adjacent-group-menu.png)

2. Now, **Tablix Group** dialog will open like below.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/group-dialog.png)

3. Choose field in the group dialog and click **OK** button.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/choose-field-in-group-dialog.png)

4. Now, a new adjacent group will be added above or below a group in the grouping panel.

Adjacent Before

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/adjacent-above.png)

Adjacent After

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/adjacent-after.png)

Add total

You can add total above or below a group. To add total click on the **Before** or **After** under **Add Total** menu.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/add-total-menu.png)

The total row or column will be added in the tablix region for the respective group in the tablix report item in design area.

Delete Group

To delete a group using grouping panel, click on the icon in the right corner of the row or column group member. Click on **Delete Group** option in the menu.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-group-menu.png)

Now, the following confirmation dialog will be launched.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-group-dialog.png)

- Choose the **Delete group and related rows and columns** option to delete all the rows and columns associated with the respective group.
- Choose the **Delete group only** option to delete the group alone.

Tablix member properties

The properties are listed under different categories in the properties panel based on the behaviour of each properties of tablix member.

Static member properties

Click on the required static tablix member in the **Row Groups** or **Column Groups** pane, now the respective tablix member properties will be listed in the properties panel.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/open-static-member-properties.png)

Miscellaneous

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/static-member-miscellaneous.png)

Fixed Data

Fixed data property is used to keep the row or column headers visible when scrolling the pages of a report in Bold Reports Report Viewer.

- To make row header visible when scrolling the pages of report, select the static header member in **Row Groups** pane and enable the **Fixed Data** property in the properties panel.

- To make column header visible when scrolling the pages of report, select the static header member in Column Groups pane and enable the Fixed Data property in the properties panel.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/enable-fixed-data.png)

Keep Together

Keep together property can be used to display the entire tablix member and any nested members in a single page, on report preview or export action. Enable the checkbox to keep the content in single page or it will span across multiple pages.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/enable-keep-together.png)

Keep with group

Keep with group property can be used to keep the row with the previous or following sibling group member.

- **None** - Select this option to indicate no preference for keeping this member with the members of the selected row group.
- **Before** - Select this option to keep the respective member with the members of the previous group.
- **After** - Select this option to keep the respective member with the members of the following group.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/keep-with-group-property.png)

Repeat on new page

Repeat on new page property can be used to display the static row or column headers on multiple pages in a report.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/repeat-on-new-page.png)

Visibility

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/visibility-property.png)

Visibility - Visibility property is used to conditionally show or hide the row or column when the report is initially run. To set visibility of static row or column using properties panel refer [Visibility](#) section.

Toggle - The toggle settings property can be used to allow the user interactively expand or collapse the report items or row and columns associated with the group to drill down to further detail within the same report. Some common reasons to use the toggle visibility feature are as follows:

- To hide columns or rows with details in table and matrix report items
- To completely hide a table or matrix item
- To hide other report items

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/toggle-property.png)

Group member properties

Click on the required group tablix member in the **Row Groups** or **Column Groups** pane, now the respective tablix member properties will be listed in the properties panel.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/open-group-member-properties.png)

Basic settings

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/group-member-basic-settings.png)

Groups

You can edit the row or column group using this option. Select a row or column group, click on the **Set Groups...** button in the properties panel. Now, the **Grouping** dialog will be opened like below.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/group-dialog.png)

Now, modify the group **Name** or group expression in the group dialog and click **OK**.

Filters

Filters can be used to filter the data in the tablix. In tablix data region, filters can be applied independently for row groups, column groups, and adjacent groups. Select a group in the grouping panel, click on the **Set Filters...** button in the properties panel. Now, the filter dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/filters-dialog.png)

To create a filter expression for the respective group member, refer the steps provided in [Filter Data](#) section.

Sorts

To sort the numeric or string field in the tablix, sorting can be used. In tablix, the sorting can be applied to the whole data region or for each group, including the details group. Select a group in the grouping panel, click on the **Set Sorts...** button. Now, the sort dialog will be opened like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/sort-dialog.png)

To create a sort expression for the respective group member, refer the steps provided in [Sort Data](#) section.

Miscellaneous

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/group-member-miscellaneous.png)

Fixed Data

Fixed data property is used to keep the row or column group headers visible when scrolling the pages of a report in Bold Reports Report Viewer.

- To make row header visible when scrolling the pages of report, select the dynamic header member in **Row Groups** pane and enable the **Fixed Data** property in the properties panel.
- To make column header visible when scrolling the pages of report, select the dynamic header member in **Column Groups** pane and enable the **Fixed Data** property in the properties panel.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/enable-fixed-data.png)

Keep Together

Keep together property can be used to display the entire tablix member and any nested members in a single page, on report preview or export action. Enable the checkbox to keep the content in single page or it will span across multiple pages.

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/enable-keep-together.png)

Page break

The page break property is used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for dynamic member.

1. Enable the **Enable Page Break** property checkbox.

![Page break property](/static/assets/on-premise/images/report-designer/report-items/tablix/group-member-page-break.png)

2. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

3. To reset page number on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

Visibility

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/visibility-property.png)

Visibility - Visibility property is used to conditionally show or hide the row or column groups when the report is initially run. To set visibility of dynamic row or column members using properties panel refer [Visibility](#) section.

Toggle - The toggle settings property can be used to allow the user interactively expand or collapse the report items or row and columns associated with the group to drill down to further detail within the same report. Some common reasons to use the toggle visibility feature are as follows:

- To hide columns or rows with details in table and matrix report items
- To completely hide a table or matrix item
- To hide other report items

![Open static member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/toggle-property.png)

Assign data to tablix data region

The table, matrix and list report items can display data from a single dataset. You can also assign conditional expressions to filter or sort the dataset columns.

Assign dataset

There are two ways to assign the dataset to the tablix:

- Assign dataset from data assign menu
- Assign dataset in **Dataset** property of tablix

Assign data from data assign menu

1. Select any cell in the tablix data region. Click on the **Data assign** menu icon to open data assign menu. If the report has no dataset, then the data assign menu will have following options.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/data-assign-menu-without-dataset.png)

Click on the **Add Datasource** option in the menu and add dataset using the steps provided in [Create Data](#) section.

2. By default, the menu displays the fields of the first dataset in the report.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/open-menu-to-choose-dataset.png)

3. The available datasets in the report will be listed in the first dropdown. To assign different dataset, choose desired dataset in the dropdown.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/dataset-dropdown-view-in-menu.png)

4. Now, the fields of selected dataset will be listed in the data assign menu like below.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/assign-desired-dataset-using-dropdown-in-menu.png)

Assign data from properties panel

1. Select the tablix report item in design area, now the respective tablix properties will be listed in the properties panel.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/table-item-with-properties-view.png)

2. The available datasets in the report will be listed in the **Dataset** property dropdown. You can choose the desired dataset from the drop-down.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/assign-desired-dataset-using-dropdown-in-panel.png)

3. Now, the fields of selected dataset will be listed in the data assign menu like below.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/fields-listed-in-data-assign-menu.png)

Assign fields to tablix cell

There are a few different ways to assign the fields into a tablix cell:

- Drag and drop data fields from the DATA panel.
- Select the field from the data assign menu provided in each cell.
- Go to the textbox's properties and assign field in Content property.

Drag and drop data fields

1. Open the DATA panel and expand the required dataset.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/data-list-view.png)

2. Drag a field from the list and drop into a required cell.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-field-into-cell.png)

Assign fields using data assign menu

1. Select the tablix cell and click on the Data assign menu icon to open data assign menu.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/data-assign-menu-icon.png)

2. Click on the required data field name in the menu.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/open-data-assign-menu.png)

The fields of the dataset which is assigned to Dataset property of tablix will be listed in the menu.

3. Now, the respective field will be assigned to the respective cell like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/assign-field-in-table-cell-output.png)

Assign data to tablix data region

Assign or edit expression into table cell

Textbox properties

1. Select a cell in the tablix, now the respective textbox properties will be listed in the properties panel.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/selected-cell-properties.png)

2. In the **Content** property, type the value or set the data field using the expression editor.

Refer [Set expression](#) section to open expression menu and to set expression.

3. Here, the **=Fields!Sales.Value** expression is entered in the content property field to assign **Sales** data field into the respective cell.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/enter-field-value-in-content-property.png)

4. Now, the **Sales** field will be denoted in the respective cell like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/assign-field-in-content-property-output.png)

Assign or edit expression into table cell

There are two ways to assign or edit the expression into a tablix cell:

- Set expression using the **Add Expression** and edit expression using the **Edit Expression** option in data assign menu
- Set or edit expression in **Content** property of a cell in properties panel

Set expression using data assign menu

1. Select a table cell and click on the **Data assign** menu icon to open data assign menu.

![Add Expression option in data assign menu](/static/assets/on-premise/images/report-designer/report-items/tablix/add-expression-option-data-assign-menu.png)

2. Click the **Add Expression** option in the data assign menu. Now, the **Expression** builder will be launched like below.

![Add Expression option in data assign menu](/static/assets/on-premise/images/report-designer/report-items/tablix/expression-builder-for-add-expression-menu.png)

To learn more about handling expressions in report designer refer [Expressions](#) section.

3. You can specify dataset fields, parameters or any other built-in functions and click on the **OK** button.

4. The label denoted in the table cell varies based on the assigned values in table cell. The following snap depicts the label text variations.

![Label text variation](/static/assets/on-premise/images/report-designer/report-items/tablix/text-representation-in-cell.png)

[Set expression in properties panel](#)

1. Select a cell in the tablix, now the respective textbox properties will be listed in the properties panel.
2. Click on the square icon in the right corner of the **Content** property.

![Label text variation](/static/assets/on-premise/images/report-designer/report-items/tablix/open-expression-menu-for-content-property.png)

3. Click on the **Expression** option in the menu. Now, the **Expression builder** will be launched like below.

![Add Expression option in data assign menu](/static/assets/on-premise/images/report-designer/report-items/tablix/expression-builder-for-add-expression-menu.png)

4. You can specify dataset fields, parameters or any other built-in functions and click on the **OK** button.
5. The label denoted in the table cell varies based on the assigned values in table cell. The following snap depicts the label text variation.

![Label text variation](/static/assets/on-premise/images/report-designer/report-items/tablix/text-representation-in-cell.png)

[Edit expression using data assign menu](#)

1. Select a table cell in which you need to edit the expression and click on the **Data assign** menu icon to open data assign menu.

![Edit expression option in data assign menu](/static/assets/on-premise/images/report-designer/report-items/tablix/edit-expression-option-in-data-assign-menu.png)

2. Click the **Edit Expression** option in the data assign menu. Now, the **Expression builder** will be launched like below.

![Label text variation](/static/assets/on-premise/images/report-designer/report-items/tablix/edit-expression-in-expression-builder.png)

3. Modify the expression as required and click on the **OK** button.

Edit expression in properties panel

1. Select a table cell, now the respective textbox properties will be listed in the properties panel.
2. Click on the square icon in the right corner of the Content property.

![Edit expression in properties panel](/static/assets/on-premise/images/report-designer/report-items/tablix/edit-expression-option-in-properties-panel.png)

3. Click on the Expression option in the menu. Now, the Expression builder will be launched like below.

![Label text variation](/static/assets/on-premise/images/report-designer/report-items/tablix/edit-expression-in-expression-builder.png)

4. Modify the expression as required and click on the OK button. Now, the expression will be updated for respective cell.

Tablix cell

Tablix cell contains the actual content to be presented by the table.

Change an item within a cell

A tablix cell contains the report items to display the data in the tabular format. By default, all cells in tablix contains the Textbox report item. You can place any of the other report items supported in Web Report Designer inside the tablix cell to present data. There are two ways to change an item within a cell.

- Drag and drop report item into table cell
- Insert item using cell menu

Drag and drop report item into table cell

Drag and drop the required report item into the table cell from item panel.

![Drag and drop line into table cell](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-image-into-table-cell.png)

Now, the item will replace the existing textbox in the cell and inserts a line report item.

![Drag and drop image into table cell](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-image-into-table-cell-output.png)

Similarly, you can insert other report items into the table cell. The following snap shows the tablix cells with different report items

![Drag and drop image into table cell](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-cell-with-different-reprt-items.png)

When you drag and drop an item into the cell which contains text box, line, image, sub-reports, chart or barcode report item, the old item in the cell will be replaced by a new report item. If the cell contains a report items such as a rectangle, list, table, or matrix, the new item is added to the containing item instead of replacing it. To replace such report item with a new item, delete the respective item from the

cell. Deleting the rectangle, list, table, or matrix item replaces it with a text box, which you can then replace with another item.

Insert item using cell menu

1. Select a cell in tablix and right click in the respective cell. Now, the cell menu will be opened like below.

![Drag and drop image into table cell](/static/assets/on-premise/images/report-designer/report-items/tablix/open-cell-menu.png)

2. Choose the required report item from the **Insert** menu option.

![Drag and drop image into table cell](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-report-item-menu-items.png)

3. Now, the old report item in the cell will be replaced by new report item.

![Drag and drop image into table cell](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-image-into-table-cell-output.png)

Here, line report item is used to demonstrate the behaviour.

When you insert item into the cell using the **Insert** option from cell context menu, the report item in the cell will be replaced by a new report item. To insert items inside the container report item such as a rectangle, drag and drop the required report item into the tablix cell containing rectangle.

Cell properties

The cell properties is properties of the report item placed in the respective cell. For example, if the cell contains the line report item, then the line report item properties will be displayed in the properties panel. To style a tablix cell, select the required cell and modify the respective report item properties in the properties panel.

Refer the [Properties panel](#) section to set and edit properties report item properties.

Delete an item from a cell

To delete the item from the tablix cell, select cell and right click. Click on **Delete** option.

![Delete cell menu](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-cell-menu.png)

Now, a new **Textbox** will be inserted in the respective cell.

![Delete cell menu](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-cell-menu-output.png)

You can also, delete the item from cell using **Delete** option in the toolbar. Select the cell and click on the delete icon in the toolbar.

![Delete cell menu](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-item-from-cell-using-delete-option.png)

Cut Copy and Paste cell contents

To cut, copy and paste cell contents in the table cell make use of the right-click menu or use the corresponding icons available in the toolbar,

- Cut - Select the cell content and use the **Cut** icon in the toolbar or right-click and select the **Cut** option from the menu to delete the selected cell content. The cut data can be later inserted in any other cell in the tablix or in the design area.
- Copy - Select the cell content and either use the **Copy** icon in the toolbar or right-click and select the **Copy** option from the menu. The copied cell content can be later inserted in any other cell in the tablix or in the design area.
- Paste - Select a cell and either use the **Paste** icon in the toolbar or right-click and select the **Paste** option to insert the previously copied/cut cell content to the selected cell or in the design area.

You can also use the following keyboard shortcuts to perform cut, copy and paste operations.

- Ctrl+X key combination for cutting;
- Ctrl+C key combination for copying;
- Ctrl+V key combination for pasting.

Cut Copy and Paste operation in single cell

Cut :

1. Select a tablix cell, use the **Cut** icon in the toolbar or right-click and select the **Cut** option from the menu.

![Cut menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/cut-option-in-menu-and-toolbar.png)

2. Now, the report item will be deleted and new **Textbox** will be inserted in the respective cell.

![Cut menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/cut-action-output.png)

Copy :

1. Select a tablix cell, use the **Copy** icon in the toolbar or right-click and select the **Copy** option from the menu.

![Copy menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/copy-option-in-menu-and-toolbar.png)

2. The copied cell content can be later inserted in any other cell in the tablix or in the design area.

Paste :

1. Select a tablix cell, use the **Paste** icon in the toolbar or right-click and select the **Paste** option in the cell menu.

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/paste-option-in-menu-and-toolbar.png)

2. Now, previously copied item will be pasted in the selected cell.

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/paste-single-cell.png)

Cut Copy and Paste operation in multiple cell

You can perform the cut/copy and paste action in multiple cells at a time.

1. Select the required cells in the tablix data region and perform copy or cut action. Here, copy action is performed.

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/multiple-cell-selection-for-copy-action.png)

2. Now, select the cells in which you need to paste the copied cell content in the tablix data region and perform paste action.

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/perform-paste-action.png)

3. The copied cell contents will be pasted in the selected cells like below,

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/paste-multiple-cell-output.png)

The copied cell content area and target cell content area to perform paste action need to be same. In the below example, the cell content in the second row, first two columns are copied and trying to pasting the cell content in second row, last column.

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/copy-multiple-cell-content.png)

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/paste-content-in-multiple-cell.png)

Now, the following alert message will be shown in the design area.

![Paste menu options](/static/assets/on-premise/images/report-designer/report-items/tablix/paste-alert.png)

So, whenever performing the paste action, the target area must have enough left, top, right, and bottom cell areas.

Tablix cell border

A cell is the intersection between a row and a column in a tablix data region. Each cells are visually separated using the border based on the collapsed-border table rendering model. Two adjacent cells will share a border.

Border behaviour

In the below snap, the table item contains three row and columns, the border behaviour for each cell varies based on its position in the table.

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/tablix-design.png)

If we apply border for the cell in position (0,0) of the table, the border will be applied to the respective cell based on the collapsing borders model behaviour.

| Border Type | Applicable |
|---------------|------------|
| border-left | Yes |
| border-top | Yes |
| border-right | Yes |
| border-bottom | No |

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/border-first-row-first-cell.png)

If we apply border for the cell in position (0,1) of the table,

| Border Type | Applicable |
|---------------|------------|
| border-left | No |
| border-top | Yes |
| border-right | Yes |
| border-bottom | No |

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/border-first-row-second-cell.png)

The other cells in the first row inherits the same behaviour as (0,1) cell.

Except last row of the table, other rows in the table inherits the above border behaviour.

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/common-border-behaviour.png)

If we apply border for last row, first cell position,

| Border Type | Applicable |
|-------------|------------|
| border-left | Yes |

Resize tablix data region

Over all data region

|border-top|Yes|

|border-right|Yes|

|border-bottom|Yes|

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/border-last-row-first-cell.png)

If we apply border for last row, second cell position,

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/border-last-row-second-cell.png)

| Border Type | Applicable |
|----------------|------------|
| border-left No | |

|border-top|Yes|

|border-right|Yes|

|border-bottom|Yes|

The other cells in the last row inherits the same behaviour as second cell position.

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/last-row-border-behaviour.png)

On report preview, the border properties will be applied to the four sides of each cell in the table.

![Table design](/static/assets/on-premise/images/report-designer/report-items/tablix-cell-border/preview.png)

Resize tablix data region

To improve the report readability, we can resize the height and width of specific column and row or the over all tablix data region.

Over all data region

Resize using resizer

Change height property:

To change the height of the tablix region , place the mouse pointer in the CenterTop or CenterBottom position of the tablix selection line.

Use CenterBottom to increase height towards downward direction or to decrease height towards upward direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-data-region-vertically.png)

Use CenterTop to increase height towards upward direction or to decrease height towards downward direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-data-region-vertically-upwards.png)

Change width property:

To change the width of the tablix region , place the mouse pointer in the RightCenter or LeftCenter position of the tablix selection line.

Use RightCenter to resize the width outwards or inwards in right side direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-data-region-right-center.png)

Use LeftCenter to resize the width outwards or inwards in left direction,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-data-region-left-center.png)

Change width and height proportionally:

1. Select the tablix data region in the design area.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/select-tablix-data-region-to-resize.png)

2. To resize the data region place the mouse pointer in LeftTop or RightTop or LeftBottom or RightBottom position on the data region. Now, the resizer arrow will be enabled in the respective position like below,

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-bottom-label.png)

3. Hold and drag the resizer arrow, now the tablix data region will be resized proportionally in all direction.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/over-all-resize-output.png)

Set width and height in table properties

1. Select a tablix data region, now the respective item properties will be listed in the properties panel.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/select-tablix-data-region-to-resize.png)

2. In the properties panel, modify the Height and Width property of the tablix data region.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-width-and-height-properties.png)

Resize the column

Set column width using gripper

1. Place the mouse pointer in the respective column border.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-column.png)

2. Drag the column gripper horizontally, to adjust the column width.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-column-output.png)

Set column width in cell properties

1. Select a cell in the tablix data region, now the respective cell item properties will be listed in the properties panel.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/select-cell-to-resize-width.png)

2. In the properties panel, modify the **Width** property of the respective cell.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/set-width-property.png)

Using touch resizer

To increase or decrease the column using touch resizer, Tap on the respective column gripper in the screen. Now, a bubble will be enabled in the right side of the column like below.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/touch-resizer-column.png)

Now, drag the bubble horizontally to adjust the column width.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/touch-resizer-column-output.png)

Resize the row

Set row height using gripper

1. Place the mouse pointer in the respective row border.

![Resize the table row](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-row.png)

2. Drag the row gripper vertically upwards or downwards, to adjust the row height.

![Adjust row height of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-row-output.png)

Set row height in cell properties

1. Select a cell in the tablix data region, now the respective cell item properties will be listed in the properties panel.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/select-cell-to-resize-width.png)

2. In the properties panel, modify the **Height** property of the respective cell.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/set-height-property.png)

Using touch resizer

To increase or decrease the row height using touch resizer, Tap on the respective row gripper in the screen. Now, a bubble will be enabled in the bottom of the row like below.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/touch-resizer-row.png)

Now, drag the bubble vertically to adjust the row height.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/touch-resizer-row-output.png)

Insert or Delete a Row

You can insert or delete rows in the tablix to organize data in your report in a better way.

Insert a row

You can add a row in two ways:

By selecting respective row,

1. Right-click on a row gripper where you want to insert a row.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/right-click-and-open-menu-to-insert-row.png)

2. Click on **Insert Row** and then click **Above** or **Below**.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-menu-options.png)

By selecting a respective cell,

1. Right-click a cell in the tablix where you want to insert a row.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/open-cell-menu-to-insert-row.png)

2. Click on **Insert Row** and then click **Above** or **Below**.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-menu-options-in-cell.png)

Now, a new row will be added above or below of the target row.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-output.png)

Insert a row in a group

You can add a row in a group in two ways:

By selecting respective row,

1. Right-click on a row group gripper where you want to insert a row.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/right-click-and-open-menu-to-insert-row-in-a-row-group.png)

2. Click on **Insert Row**.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-in-a-row-group-menu-options.png)

By selecting a respective cell,

1. Right-click a row group cell in the tablix where you want to insert a row.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/open-cell-menu-to-insert-row-in-a-group.png)

2. Click on **Insert Row**

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-in-group-menu-options-in-cell.png)

The following options will be listed in the **Insert Row** menu.

1. **Inside Group - Above** - A new row is added inside the target group but in the top position. In the below snap, the **Grouping Indicator** denotes that the row is added inside the group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-inside-group-above.png)

2. **Inside Group - Below** - A new row is added inside the target group but in the bottom position. In the below snap, the **Grouping Indicator** denotes that the row is added inside the group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-inside-group-below.png)

3. **Outside Group - Above** - A new row is added outside of the target group but in the top position.
In the below snap, notice that the **Grouping Indicator** is shown only for the target group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-outside-group-above.png)

4. **Outside Group - Below** - A new row is added outside of the target group but in the bottom position. In the below snap, notice that the **Grouping Indicator** is shown only for the target group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-row-outside-group-below.png)

Delete a row

You can delete a row in two ways:

By selecting respective row,

1. Select a row that you want to delete.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/select-a-row-to-delete.png)

2. Right-click on the row gripper of the respective row.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/open-delete-menu-of-selected-row.png)

3. Then click on the **Delete Rows** in the context menu.

By selecting a respective cell,

1. Right-click a cell in the tablix where you want to delete a row.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/open-delete-menu-of-selected-cell.png)

2. Then click on the **Delete Rows** in the context menu.

Now, a selected row or rows will be deleted from the tablix.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-row-output.png)

Delete a row from a group

1. Right-click a row group cell in the row group of a tablix.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/open-delete-menu-of-selected-cell-row-group.png)

2. Then click **Delete Rows**.

Now, a selected row will be deleted from the tablix.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-row-output.png)

Insert or Delete a Column

You can insert or delete columns in the tablix to organize data in your report in a better way.

Insert a column

You can add a column in two ways:

By selecting respective column,

1. Right-click on a column gripper where you want to insert a column.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/right-click-and-open-menu-to-insert-column.png)

2. Click on **Insert Column** and then click **Left** or **Right**.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-menu-options.png)

By selecting a respective cell,

1. Right-click a cell in the tablix where you want to insert a column.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/open-cell-menu-to-insert-column.png)

2. Click on **Insert Column** and then click **Left** or **Right**.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-menu-options-in-cell.png)

Now, a new column will be added above or below of the target column.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-output.png)

Insert a column in a group

You can add a column in a group in two ways:

By selecting respective row,

1. Right-click on a column group gripper where you want to insert a column.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/right-click-and-open-menu-to-insert-column-in-a-column-group.png)

2. Click on **Insert Column**.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-in-a-column-group-menu-options.png)

By selecting a respective cell,

1. Right-click a column group cell in the tablix where you want to insert a column.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/open-cell-menu-to-insert-column-in-a-group.png)

2. Click on **Insert Column**

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-in-group-menu-options-in-cell.png)

The following options will be listed in the **Insert Column** menu.

1. **Inside Group - Left** - A new column is added inside the target group but in the left position. In the below snap, the **Grouping Indicator** denotes that the column is added inside the group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-inside-group-left.png)

2. **Inside Group - Right** - A new column is added inside the target group but in the right position. In the below snap, the **Grouping Indicator** denotes that the column is added inside the group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-inside-group-right.png)

3. **Outside Group - Left** - A new column is added outside of the target group but in the left position. In the below snap, notice that the **Grouping Indicator** is shown only for the target group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-outside-group-left.png)

4. **Outside Group - Right** - A new column is added outside of the target group but in the right position. In the below snap, notice that the **Grouping Indicator** is shown only for the target group.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/insert-column-outside-group-right.png)

Delete a column

You can delete a column in two ways:

By selecting respective column,

1. Select a column that you want to delete.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/select-a-column-to-delete.png)

2. Right-click on the column gripper of the respective column.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/open-delete-menu-of-selected-column.png)

3. Then click on the **Delete Columns** in the context menu.

By selecting a respective cell,

1. Right-click a cell in the tablix where you want to delete a column.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/open-delete-menu-of-selected-cell-to-delete-column.png)

2. Then click on the **Delete Columns** in the context menu.

Now, a selected column will be deleted from the tablix.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-column-output.png)

Delete a column from a group

1. Right-click a column group cell in the column group of a tablix.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/open-delete-menu-of-selected-cell-column-group.png)

2. Then click **Delete Columns**.

Now, a selected column will be deleted from the tablix.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-column-group-ooutput.png)

Tablix Group

A group organizes the report dataset in a data regions. It helps to present the different views of the same data in a data regions.

The following snap shows the areas for a tablix region with nested row groups and column groups

![Tablix Groups](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-areas.png)

Groups are organized as members of one or more hierarchies for each data region. A group hierarchy has parent/child groups that are nested and can have adjacent groups. Data associated with row group members expands horizontally across the page and data associated with column group members expands vertically down the page. The **Grouping panel** displays row group and column group members for the currently selected tablix data region on the design surface.

Create groups

A group has a name and a set of group expressions. The set of group expressions can be a dataset field or a combination of multiple expressions. During the data processing, the group expressions are combined and the resultant data is applied to the specific group. For example, if a group has

date field to organize the data in the data region. On report preview, data is organized by date, and then displayed with other dataset values for each date.

By default, a table, matrix, or list, report items created with **Details** group only. You can create groups manually to organize and group data in the tablix data regions.

Group types

- **Row groups and column groups** - The data can be organized into groups by rows or columns. On report preview, the row groups expand vertically on a page and the column groups expand horizontally on a page. You can create a nested group or adjacent groups in the data regions. When you create a group for a data region, set of rows or columns will be added to the data region and these rows or columns are used to display group data. Refer [Row Groups](#) and [Column groups](#) section to create groups in tablix data region.
- **Details Group** - The Details group displays all data from the dataset after applying the dataset and data region filters. The **Details** group is the only group that has no group expression. By default, a table or list report item is created with the Details group and adds a row to display the detail data. On report preview, the details row repeats once for every value in the data which is applied to the respective data region. Refer [Create Details Group](#) section to create details group in tablix data region.

A details group is added as a child group in the Grouping panel, you cannot add child group to the details group further.

Edit group properties

After creating groups, the group properties such as [filters](#), [sort](#), and [groups expressions](#) can be modified to organize the data visualization of the respective group in data region. To edit an existing group, select

the respective member in the Row or Column Groups pane of the grouping panel. Now, the respective member properties will be listed in the properties panel.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/open-group-member-properties.png)

Refer [Group properties](#) section and modify the group properties.

Group scope

You can create a nested group or adjacent groups in the data regions. The scope of group is defined by the hierarchy in which the groups are created in the data regions. To easily understand the scope of the groups in the selected tablix data region,

- The visual cues are provided in the tablix data regions.

![Groups and total](/static/assets/on-premise/images/report-designer/report-items/tablix/groups-and-total-sketch.png)

- The groups are listed as tree structure in the grouping panel.

![Group types sketch](/static/assets/on-premise/images/report-designer/report-items/tablix/group-types-sketch.png)

To understand the visual cues in grouping panel, refer the [Grouping Panel Visual Cues](#) section.

Insert or Delete a Row Group

You can add row groups in the table to display data in a visual hierarchy. You can create both nested groups and adjacent groups. Row group can be added either using the cell context menu or using the group menu provided in grouping panel. In cell context menu the **Row Group** option will be shown only if there is a possibility to add row group in the respective place.

Insert a row group

In tablix data region the data can be grouped using the following options:

- Parent Group
- Child Group
- Adjacent Group

The parent and child group are used for creating a hierarchy, the groups depend on each other. While the adjacent grouping functionality can be used to create separate groupings in the same table.

Parent row group

To add a first row group in the basic tablix structure follow the below steps.

1. Select the tablix data region in the design area, now the **Grouping Panel** will be enabled in the design view.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/select-tablix-data-region-in-design-area.png)

To add first row group in the basic tablix structure, use the group menu in grouping panel.

2. To add a group, go to **Row Groups** pane in grouping panel and open the context menu on the **Details** group field.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/open-context-menu-in-details-group.png)

3. From the context menu, click on **Parent Group...** option under **Add Group** category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/click-on-parent-group-option.png)

4. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

- **Group By:** Based on the dataset assigned to the tablix region, dataset fields will be listed in this drop-down or else click on the square icon to create an expression.
 - **Add Group Header:** Enable this option to add a header to this group
 - **Add Group Footer:** Enable this option to add a footer to this group
5. Here, **Product Category** field is chosen as parent group in tablix data region. Click on the **OK** button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/assign-field-for-parent-group.png)

Now, a new column will be added to the right side of the detail group in tablix data region and a new group member will be added above the **Detail** group of **Row Groups** pane in grouping panel hierarchical view.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/add-parent-group-in-row-group.png)

On report preview, each product sales details will be grouped based on product category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/row-group-output.png)

Child row group

To sub categorize the data in tablix data region you can add child group to the tablix. Now, to add the child group we can either use the cell menu or group menu in grouping panel. In the following steps child group is added using cell menu.

1. Select and right click in the row group cell to which you need to add a child group.
2. In the cell menu, click on the **Child Group** under the **Row Group** category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/open-cell-menu-to-add-child-group.png)

3. Once you click on the **Child Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

- **Group By:** Based on the dataset assigned to the tablix region, dataset fields will be listed in this drop-down or else click on the square icon to create an expression.
- **Add Group Header:** Enable this option to add a header to this group
- **Add Group Footer:** Enable this option to add a footer to this group
 4. Here, **Sub Category** field is chosen as child group in tablix data region. Click on the **OK** button.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/assign-field-for-child-group.png)

Now, a new column will be added to the right side of the **Product Category** row group in tablix data region and a new field will be added below of the **Product Category** group in hierarchical view of **Row Groups** pane in grouping panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/add-child-group-in-row-group.png)

On report preview, product names will be categorized based on the product type.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/child-row-group-output.png)

To add child group using group menu in grouping panel, refer [Add child group](#) section.

Adjacent row group

To create multiple groupings in a tablix data region adjacent grouping can be used. The adjacent row group can be added above/before or below/after of the existing group in the data region. Now, to add the adjacent group we can either use the cell menu or group menu in grouping panel. Follow the below steps to add a adjacent group.

1. Select and right click in the a row group cell to which you need to create a adjacent group.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/open-adjacent-group-menu.png)

2. Click on **Adjacent Above** to create a new group before the existing group or **Adjacent Below** to create a new group after the existing group.
3. Once you click on the **Adjacent Above** or **Adjacent Below** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

- Choose the required dataset field in the **Group by** field and click on the **OK** button.

Now, a new row group will be created above or below the existing group in the tablix data region.

To add adjacent group using group menu in grouping panel, refer [Add adjacent group](#) section.

Delete row group

Row group can be deleted either using the cell context menu or using the group menu provided in grouping panel.

- Select and right click on a row group which you want to delete from the tablix data region.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/delete-row-group-menu.png)

- In the cell menu, click on the **Delete Row Group** option. Now, the following confirmation dialog will be launched.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/delete-row-group-dialog.png)

- In the **Delete Group** dialog choose one of the following options and click on the **OK** button.
 - Delete group and related rows and columns** - Choose this option to delete the group and all related rows that display group data. For the details group, if the same row belongs to both detail and group data, only the detail data rows are deleted.
 - Delete group only** - Choose this option to keep the structure of the tablix data region and delete only the group definition.

Now, the respective group information will be deleted from the tablix data region.

Edit group properties

To edit a row group properties in a tablix data region, refer [Tablix member properties](#) section.

Insert or Delete a Column Group

You can add column groups in the table to display data in a visual hierarchy. You can create both nested groups and adjacent groups. Column group can be added either using the cell context menu or using the group menu provided in grouping panel. In cell context menu the **Column Group** option will be shown only if there is a possibility to add column group in the respective place.

Insert a column group

In tablix data region the data can be grouped using the following options:

- Parent Group
- Child Group
- Adjacent Group

The parent and child group are used for creating a hierarchy, the groups depend on each other. While the adjacent grouping functionality can be used to create separate groupings in the same table.

[Parent column group](#)

To add a first column group in the basic tablix structure follow the below steps.

1. Select and right click in the cell to which you need to add a parent child group.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/open-column-group-cell-menu.png)

2. In the cell menu, click on the **Parent Group** under the **Column Group** category.
3. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

- **Group By:** Based on the dataset assigned to the tablix region, dataset fields will be listed in this drop-down or else click on the square icon to create an expression.
 - **Add Group Header:** Enable this option to add a header to this group
 - **Add Group Footer:** Enable this option to add a footer to this group
4. Here, **OrderQtr** field is chosen as parent group in tablix data region. Click on the **OK** button.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/assign-field-for-column-parent-group.png)

Now, a new row will be added to the above the target cell in tablix data region and a new group member will be added in the **Column Groups** pane in grouping panel hierarchical view.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/add-parent-group-in-column-group.png)

On report preview, each quarter sales of a year will be grouped along column.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/column-group-output.png)

Similarly, you can add child and adjacent group in the column group. Refer [Insert or Delete Row Group](#) section to add row groups.

[Delete column group](#)

Column group can be deleted either using the cell context menu or using the group menu provided in grouping panel.

1. Select and right click on a column group which you want to delete from the tablix data region.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/delete-column-group-menu.png)

2. In the cell menu, click on the **Delete Column Group** option. Now, the following confirmation dialog will be launched.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/delete-row-group-dialog.png)

3. In the **Delete Group** dialog choose one of the following options and click on the **OK** button.
 - o **Delete group and related rows and columns** - Choose this option to delete the group and all related rows that display group data. For the details group, if the same column belongs to both detail and group data, only the detail data rows are deleted.
 - o **Delete group only** - Choose this option to keep the structure of the tablix data region and delete only the group definition.

Now, the respective group information will be deleted from the tablix data region.

Edit group properties

To edit a column group properties in a tablix data region, refer [Tablix member properties](#) section.

Add or Delete a Details Group

The Details group can be used to display the detailed data in the tablix data region and it has no group expression. By default, when you drag and drop the the tablix report item in the design area,it renders with a **Details** group.

Add details group

1. Select the tablix data region in the design area, now the **Grouping Panel** will be enabled in the design view.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-or-delete-details-group-ssrs/enable-grouping.png)

2. To add a details group, go to **Row Groups** pane in grouping panel and open the context menu on the innermost child group. Click **Add Group**, and then click on the **Child Group** option.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-or-delete-details-group-ssrs/add-child-group-menu.png)

3. Once you click on the **Child Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-or-delete-details-group-ssrs/tablix-group-dialog.png)

4. Select **Show detail data** and click on the **OK** button.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-or-delete-details-group-ssrs/enable-show-detail-data.png)

A details group is added as a child group in the Grouping panel, so you cannot add child group to the details group further.

Delete details group

Details group can be deleted either using the cell context menu or using the group menu provided in grouping panel.

1. Click on the icon in the right corner of the Details group member. Click on Delete Group option in the menu.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-or-delete-details-group-ssrs/delete-group.png)

2. Now, the following confirmation dialog will be launched.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/delete-group-dialog.png)

- Choose the Delete group and related rows and columns option to delete all the rows and columns associated with the respective group.
- Choose the Delete group only option to delete the group alone.

Now, the respective Details group information will be deleted from the tablix data region.

Edit group properties

To edit a group properties in a tablix data region, refer [Tablix member properties](#) section.

Merge Cells

In tablix data regions, two or more adjacent cells can be merged into a single cell to improve the appearance or to provide spanning labels for column groups and row groups.

Merge restrictions

The cells in the each area of a data region can be merged within the respective areas. Following are the some of the restrictions when merging the cells in tablix data region:

- Cannot merge a cell in the corner area of data region with a cell in the row group area. Cross boundary merge is not allowed.
- In body area, the cells can only be merged in the horizontal direction and not in vertical direction.
- In corner area, the cells can be merged in horizontally across columns or vertically down rows at a time.

Merge cells

In the following sections, merge action for each area of tablix data region is demonstrated using company sales report.

![Tablix areas](/static/assets/on-premise/images/report-designer/report-items/tablix/merge-split-sketch.png)

Corner area

1. Select the cells in the corner area of the tablix data region.
2. Right click in the cell to open cell menu, in the menu list click on the Merge Cells option.

If the cell selection is invalid or across boundaries, Merge Cells option will not be available in the menu.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/merge-corner-cells.png)

3. Now, the selected cell in the corner area will be merged like below.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/corner-cell-merge-output.png)

If the cell has any content, the content of the top-left cell will be displayed in the newly merged cell. The content of the rest of the cells in the merged region will be cleared.

Body area

1. Select the cells in the body area of the tablix data region.
2. Right click in the cell to open cell menu, in the menu list click on the Merge Cells option.

If the cell selection is invalid or across boundaries, Merge Cells option will not be available in the menu.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/select-cell-in-body-area-for-merge-action.png)

3. Now, the selected cell in the body area will be merged like below.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/body-cells-merge-output.png)

If the cell has any content, the content of the top-left cell will be displayed in the newly merged cell. The content of the rest of the cells in the merged region will be cleared.

Row group

1. Select the cells in the row group of the tablix data region.
2. Right click in the cell to open cell menu, in the menu list click on the Merge Cells option.

If the cell selection is invalid or across boundaries, Merge Cells option will not be available in the menu.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/select-cell-in-row-group-area-for-merge-action.png)

3. Now, the selected cell in the row group area will be merged like below.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/merge-row-group-cells-output.png)

If the cell has any content, the content of the top-left cell will be displayed in the newly merged cell. The content of the rest of the cells in the merged region will be cleared.

Column group

1. Select the cells in the column group area of the tablix data region.
2. Right click in the cell to open cell menu, in the menu list click on the Merge Cells option.

If the cell selection is invalid or across boundaries, Merge Cells option will not be available in the menu.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/select-cell-in-column-group-area-for-merge-action.png)

3. Now, the selected cell in the column group area will be merged like below.

![Merge corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/merge-row-group-cells-output.png)

If the cell has any content, the content of the top-left cell will be displayed in the newly merged cell. The content of the rest of the cells in the merged region will be cleared.

Split cells

Once a cell is merged, you can split using the Split Cells option in the menu list. The cells can split in horizontal direction across columns or in vertical direction down rows.

1. Select the cells in the specific area of the tablix data region.
2. Right click in the cell to open cell menu, in the menu list click on the Split Cells option.

If the cell selection is invalid or across boundaries, Split Cells option will not be available in the menu.

![Split corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/select-cell-in-column-group-area-for-split-action.png)

3. Now, the selected cell in the column group area will split like below.

![Split corner cells](/static/assets/on-premise/images/report-designer/report-items/tablix/split-row-group-cells-output.png)

If the cell has any content, the content of the top-left cell will be displayed in the newly merged cell. The content of the rest of the cells in the merged region will be cleared.

Add filters to tablix data region

Filters can be used to filter data in the tablix data region or a data region group to include or exclude specific values from display or to provide a different view of the dataset in multiple data regions. When processing the report, the filters applied in the report parts are processed first on the dataset, and then on the data region, and then on groups.

Set filter on tablix data region

1. Select tablix data region in the design area.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/select-data-region.png)

2. In the properties panel, click on the **Set Filters...** button. Now, the filter dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/filters-dialog.png)

3. Refer [Filter Data](#) section and create required filter expression in the filter dialog and click **OK**.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/create-filter-expressions.png)

In the above image, three filter equations are created to filter the data in the data region based on **OrderYear**, **Product Category**, and **OrderQtr** data fields. In the below design, two different tablix data regions are designed using single dataset to display the quarter sales **Q1** and **Q2** against a year.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/report-design-view.png)

On report preview, using the **OrderYear** and **ProductCategory** report parameters, the sales of **Q1** and **Q2** is displayed in the tablix region based on applied filters.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/report-preview.png)

Set filter on a tablix group

1. Click on the surface of the tablix data region to open grouping panel.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/enable-grouping-panel.png)

2. Click on the required group tablix member in the **Row Groups** or **Column Groups** pane, now the respective tablix member properties will be listed in the properties panel.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/open-member-properties.png)

3. In the properties panel, click on the **Set Filters...** button. Now, the filter dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/filters-dialog.png)

4. Refer [Filter Data](#) section and create required filter expression in the filter dialog and click **OK**.

Sort data in a tablix data region

Set sort expression for tablix data region

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/filter-equation-for-group.png)

Before applying filter to the row group, on report preview the report displays sales of each quarters and its total.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/before-applying-filter-report-preview.png)

After applying filters for the respective row group, the total sales of Q1 and its total is displayed.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/after-applying-filter-report-preview.png)

Here, the filter is applied on a row group group for demonstration. Similarly, you can apply filters on column groups and detail groups.

Sort data in a tablix data region

In a tablix data region, set the sort expression for the data region or for each group, including the details group.

Set sort expression for tablix data region

1. Select tablix data region in the design area.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/select-data-region.png)

2. In the properties panel, click on the **Set Sorts...** button. Now, the sort dialog will be opened like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/sort-dialog.png)

3. Refer [Sort Data](#) section to and create required sort expression in the sort dialog and click on the **OK** button.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/sort-data-in-tablix-data-region/new-sort-expression.png)

A sort expression is created to sort **ProductCategory** fields in descending order (i.e Z-A). On report preview, the **ProductCategory** field values will be sorted in Z-A alphabetical order. Respective column is highlighted in the below snap.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/sort-data-in-tablix-data-region/sorting-report-preview.png)

Set sort expression on a tablix group

1. Click on the surface of the tablix data region to open grouping panel.

Add Total

Add total tablix body

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/enable-grouping-panel.png)

2. Click on the required group tablix member in the **Row Groups or Column Groups** pane, now the respective tablix member properties will be listed in the properties panel.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/add-filter-to-tablix-data-region/open-member-properties.png)

3. In the properties panel, click on the **Set Sorts...** button. Now, the sort dialog will be opened like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/sort-dialog.png)

4. Refer [Sort Data](#) section to and create required sort expression in the sort dialog and click on the **OK** button.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/sort-data-in-tablix-data-region/new-sort-expression.png)

Add Total

The add total option can be used to display the sum of numeric values for tablix group or over all tablix data region.

Add total tablix body

1. Select the cell which contains numeric field in the tablix data region body area.
2. Right click in the cell and click on the **Add Total** option in the menu.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/open-cell-menu.png)

3. Now, new row or column will be added outside of the respective cell in the data region.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/add-total-to-body-cell-output.png)

If the cell in body area is common for row and column group as shown in below snap, the **Row** and **Column** option will be listed in **Add Total** menu

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/row-and-column-total-menu.png)

Add total for row group

1. Select the cell in the row group area for which you want total in the tablix data region.
2. Right click in the cell and click on the **Add Total** option in the menu.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/add-total-row-group-menu.png)

3. Click on **Before** or **After** option in the cell to add total after or before a group in the tablix data region.
4. Now, new row will be added outside of the current group in the data region and then a default total is added for each numeric field in the row.
 - **Add total before:**

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/row-group-add-total-before.png)

- **Add total after:**

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/row-group-add-total-after.png)

Add total for column group

1. Select the cell in the column group area for which you want total in the tablix data region.
2. Right click in the cell and click on the **Add Total** option in the menu.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/add-total-column-group-menu.png)

3. Click on **Before** or **After** option in the cell to add total after or before a group in the tablix data region.
4. Now, new column will be added outside of the current group in the data region and then a default total is added for each numeric field in the column.
 - **Add total before:**

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/column-group-add-total-before.png)

- **Add total after:**

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/add-total-to-tablix-data-region/column-group-add-total-after.png)

Add Group Header and Footer

1. Drag and drop **Table** report item from item panel into the design area and assign dataset to the tablix data region.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/select-tablix-data-region-in-design-area-to-header.png)

2. Click on the table surface to enable the **Grouping Panel** in the design view.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/enable-grouping-panel.png)

3. Go to **Row Groups** pane in grouping panel and open the context menu on the **Details** group field.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/open-context-menu-in-details-group.png)

4. From the context menu, click on **Parent Group...** option under **Add Group** category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/click-on-parent-group-option.png)

5. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/tablix-group-dialog.png)

- **Group By:** Based on the dataset assigned to the tablix region, dataset fields will be listed in this drop-down or else click on the square icon to create an expression.
- **Add Group Header:** Enable this option to add a header to this group
- **Add Group Footer:** Enable this option to add a footer to this group
 - 6. Choose dataset field in the **Group by** drop-down list.
 - 7. Select **Add header** to add header row to the group and select **Add footer** to add footer row to the group. Click on the **OK** button.
- **Add header** - Adds a static row above the group.
- **Add footer** - Adds a static row below the group.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/enable-header-and-footer-in-group-dialog.png)

Now, a static row will be added above and below of the group in the tablix data region.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/header-and-footer-output.png)

On report preview, the header and footer will be added for each group as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/header-and-footer-preview.png)

Format header and footer

You can display data, label content or total in the group header and footer of the tablix data region.

Merge header cells

1. Select the header cells and right-click in the cell. Then, click on **Merge Cells** option.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/merge-header-cells.png)

2. Now, set data or label content and format the header cell as required.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/merge-header-cells-output.png)

Merge footer cells

1. Select the first two footer cells and right-click in the cell. Then, click on **Merge Cells** option.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/merge-footer-cell.png)

2. Now, set data or label content and format the footer cell as required.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/merge-footer-cell-output.png)

Edit footer cell content

1. Edit the footer cell which is merged in the previous step and set the expression = “Total yearly sales of ” & Fields!ProdCat.Value using expression builder and set the font weight property as **Bold** in the properties panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/set-expression-in-footer.png)

2. Set =Sum(Fields!Sales.Value) expression in the last cell, to calculate summary of sales field and set the font weight property as **Bold** in the properties panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/assign-summary-field.png)

Edit header cell content

Edit the header cell and set the = “Sales Report of ” & Fields!ProdCat.Value & “ Category” expression using expression builder and set the font weight property as **Bold** in the properties panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/set-expression-in-header.png)

Report preview

On report preview, the header and footer will be added for each group as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-header-and-footer/header-and-footer-with-value.png)

Nested Data Regions

The data region such as a chart can be placed inside the another data region such as a tablix, to link more than one data region to the same dataset. This will provide different views of the same data. For example, if you want to create a sales report for each sales person, you can create a list with text boxes and an image to display information about the employee, and then add table and chart data regions to the list to show the employee's sales record.

Assign data to the nested data region

When you create a report design with nested data regions, the dataset assigned to the parent data region will be inherited by the nested data regions. Nested data regions are based on the single dataset, the data regions based on different datasets cannot be nested inside one another.

Assign data to the nested chart region

If the chart data region is nested within a table, the **Data** tab of the chart report item will be in the disabled state until you assign data to the table report item.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-data-to-chart-initial-view.png)

To enable the **Data** panel for chart report item, select the table report item and assign the dataset in the **Dataset** property using properties panel.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-data-to-table.png)

Now, select the nested chart report item in the table cell and notice the **Data** panel is in enabled state.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-data-to-chart-enabled-view.png)

Switch to the **Data** tab and assign data fields to the chart.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/chart-data-panel-full-view.png)

Assign data to the nested tablix region

If any of the tablix data region is nested within another tablix region, the **Dataset** property for the nested tablix region will not available in the tablix properties.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-data-for-nested-table.png)

So to assign data to the nested tablix region, select the parent data region and assign the dataset in the **Dataset** property using properties panel.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-data-to-parent-data-region.png)

Now, select a cell in nested tablix region and open the data assign menu. The available fields in the respective dataset will be listed in the data assign menu, from then you can assign data to the nested tablix region.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/data-assign-menu.png)

Scope of data for nested data regions

The data bind to the nested data regions, based on the position of the nested data regions in the parent data region. The following points describes the scope for cells in the following Tablix areas:

- Tablix corner - The data bind to the nested data regions, after the filter and sort expressions for the dataset and the outer Tablix are applied.
- Tablix column group - The data bind to the nested data regions in the innermost column group, after the filter and sort expressions for the dataset, the outer Tablix, and the column groups are applied.
- Tablix row group - The data bind to the nested data regions in the innermost row group, after the filter and sort expressions for the dataset, the outer Tablix, and the row groups are applied.
- Tablix body - The data bind to the nested data regions in the tablix body area, after the filter and sort expressions for the dataset, the outer Tablix, and the row and column groups are applied.

Design a simple report with nested data region

1. To present data in the data regions, create a dataset and bind data to the data region. In this designing section, the following dataset query is used for dataset creation.

```
`sql
SELECT SOD.SalesOrderDetailID, SOD.OrderQty, SOD.UnitPrice,
CASE WHEN SOD.UnitPriceDiscount IS NULL THEN 0 ELSE SOD.UnitPriceDiscount END AS
UnitPriceDiscount,
SOD.LineTotal, SOD.CarrierTrackingNumber, SOD.SalesOrderID, P.Name, P.ProductNumber
FROM Sales.SalesOrderDetail SOD INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID INNER JOIN
Sales.SalesOrderHeader SOH ON SOD.SalesOrderID = SOH.SalesOrderID
`
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

2. Drag and drop a **List** report item to the design surface.

![Add list report item](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/add-list-report-item.png)

3. Select the **List** report item and choose the dataset in the **Dataset** property using property panel.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-dataset-to-list.png)

4. In the Row Grouping pane, click on the **(Details)** field, now the tablix member properties will be listed in the properties panel.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/open-details-group-properties.png)

5. Click on the **Groups** button to open the **Grouping** dialog.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/grouping-button.png)

6. Change the Name to **SalesOrderID**. Click **Add** in the **Grouping** dialog and choose **SalesOrderID** in drop-down list and click **OK**.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/create-group-expression.png)

By adding a group expression, details group is changed as a parent group organized by sales order Id's.

7. Resize the List data region to the required size and drag and drop a textbox report item in the list report item.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/add-textboxes-inside-list-report-item.png)

8. Refer [Create Expression in Textbox](#) and assign the = “Sales Order: “ & **Fields!SalesOrderID.Value** expressions in the textbox.
9. Drag and drop a **Table** report item into List.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/add-table-in-list-report-item.png)

10. Refer [Assign data fields](#) section and assign the **ProductNumber**, **OrderQty**, **LineTotal** data fields to the table.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/assign-data-fields-in-table.png)

11. Refer [Set header text](#) section and set header text to the table header row.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/set-table-header-row-text.png)

A simple sales order details report is designed using list and table data regions to group the data based on **SalesOrderID**.

![Assign data to the list](/static/assets/on-premise/images/report-designer/report-items/tablix-nested-data-regions/final-design.png)

Design ssrs rdl report using table

The following steps guides you to design ssrs rdl report using table.

Add a table to the report

1. The table report item is listed in the item panel under the **Data Regions** category.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/item-panel-view.png)

2. Drag and drop the table report item into the design area from the item panel.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-table.png)

3. Respective table properties will be listed in the properties panel.

![Tablix item with properties view](/static/assets/on-premise/images/report-designer/report-items/tablix/table-item-with-properties-view.png)

Initial design

Once you drop the table item, the **Table** renders with two rows and three columns in the design area.

![Tablix renders with fixed number of row and columns](/static/assets/on-premise/images/report-designer/report-items/tablix/table-basic-view.png)

Each cell in the table is nothing but a simple textbox. The cell in a list contains a rectangle. You can replace a default report item with a different report item.

Assign data fields

There are a few different ways to assign the fields into a table:

- Drag and drop data fields from the **DATA** panel.
- Select the field from the data assign menu provided in each cell.
- Go to the textbox's properties and assign field in **Content** property.

Drag and drop data fields

1. Open the **DATA** panel and expand the required dataset.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/data-list-view.png)

2. Drag a field from the list and drop into a required cell.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-field-into-cell.png)

Assign fields using data assign menu

1. Select the table cell and click on the **Data assign** menu icon to open data assign menu.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/data-assign-menu-icon.png)

2. Click on the required data field name in the menu.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/open-data-assign-menu.png)

The fields of the dataset which is assigned to **Dataset** property of table will be listed in the menu.

3. Now, the respective field will be assigned to the respective cell like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/assign-field-in-table-cell-output.png)

Textbox properties

1. Select a cell in the table, now the respective textbox properties will be listed in the properties panel.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/selected-cell-properties.png)

2. In the **Content** property, type the value or set the data field using the expression editor.

Refer [Set expression](#) section to open expression menu and to set expression.

3. Here, the **=Fields!Sales.Value** expression is entered in the content property field to assign **Sales** data field into the respective cell.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/enter-field-value-in-content-property.png)

4. Now, the **Sales** field will be denoted in the respective cell like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/assign-field-in-content-property-output.png)

Set header text

You can provide the header text for each column of the table in two ways:

Using data assign menu

1. Select the table cell and click on the **Data assign** menu icon to open data assign menu.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/open-add-text-menu.png)

2. Click on **Add Text** option in the menu. Now, an **Add Text** dialog will be opened like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/add-text-dialog.png)

3. Enter the header text in the textarea and click on the **Add** button.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/enter-text-in-add-text-dialog.png)

4. Now, the text will be displayed in the respective cell like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/enter-text-in-add-text-dialog-output.png)

[Set text in content property](#)

1. Select a cell in the table, now the respective textbox properties will be listed in the properties panel.
2. In the **Content** property, type the header text in the content property textbox and press **Enter**.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/enter-text-in-content-field.png)

3. Now, the text will be displayed in the respective cell like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/enter-text-in-content-property-output.png)

[Resize the column](#)

To improve the report readability, we can resize the table row height and column width.

1. Place the mouse pointer in the respective column border.

![Resize the table column](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-column.png)

2. Drag the column gripper horizontally, to adjust the column width.

![Adjust column width of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-column-output.png)

[Resize the row](#)

1. Place the mouse pointer in the respective row border.

![Resize the table row](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-row.png)

2. Drag the row gripper vertically, to adjust the row height.

![Adjust row height of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/resize-row-output.png)

Final design

A simple table design will look like below.

![Adjust row height of the table](/static/assets/on-premise/images/report-designer/report-items/tablix/simple-table-design.png)

Add Grouping and Totals

The following steps guides you to design ssrs rdl report to add grouping and totals in tablix data region.

Create dataset

To present data in the tabular format, create a dataset and bind data to the tablix data region. In this designing section, the following dataset query is used for dataset creation.

```
'sql
SELECT Top 50 PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear,
'Q' + DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
FROM Production.ProductSubcategory PS INNER JOIN
Sales.SalesOrderHeader SOH INNER JOIN
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =
P.ProductSubcategoryID INNER JOIN
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryID
'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Simple table design

Create a simple table report design by following the steps provided in [Table Design](#) section.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/initial-design.png)

Add group data

Parent row group

To add a first row group in the basic tablix structure follow the below steps.

1. Select the tablix data region in the design area, now the **Grouping Panel** will be enabled in the design view.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/enable-grouping-panel.png)

To add first row group in the basic tablix structure, use the group menu in grouping panel.

2. To add a group, go to **Row Groups** pane in grouping panel and open the context menu on the **Details** group field.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/open-context-menu-in-details-group.png)

3. From the context menu, click on **Parent Group...** option under **Add Group** category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/click-on-parent-group-option.png)

4. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

5. Here, **Product Category** field is chosen as parent group in tablix data region. Click on the **OK** button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/assign-field-for-parent-group.png)

Now, a new column will be added to the right side of the detail group in tablix data region and a new group member will be added above the **Detail** group of **Row Groups** pane in grouping panel hierarchical view.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/add-parent-group-design-output.png)

Child row group

1. To add a child group, click on the **Child Group...** option under **Add Group** category.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/open-child-group-context-menu.png)

2. Now, **Tablix Group** dialog will open like below.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix/tablix-group-dialog.png)

3. Choose field in the group dialog and click **OK** button.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/add-child-group.png)

Now, a new column will be added to the right side of the **Product Category** row group in tablix data region and a new field will be added below of the **Product Category** group in hierarchical view of **Row Groups** pane in grouping panel.

![Open group member properties](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/add-child-group-design-output.png)

Add Totals

Add total yearly sales of a product

In the below steps the total field is created to calculate the sum of sales of each sub category products.

1. Right-click in the cell which contains **[Sales]** expression, and click on the **Add Total** in the cell menu.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/select-cell-to-add-quarterly-sales.png)

2. Now, new row will be added inside of the **ProdCat** group in the data region as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/quarterly-total-sales-new-total-row.png)

Add totals yearly sales of product category

Now to calculate the sum of yearly sales of each product category follow the below steps.

1. Right-click in the third row, last cell of the data region that contains **[Sum(Sales)]** expression, and click on the **Add Total** in the cell menu.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/yearly-total-sales.png)

2. Now, another new row will be added inside of the **ProdCat** group in the data region as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/yearly-total-sales-new-total-row.png)

Add the grand total to the report

Now to calculate the grand total of over all product sales follow the below steps.

1. Right-click in the fourth row, last cell of the data region that contains [Sum(Sales)], and click on the Add Total in the cell menu.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/total-sales.png)

2. Now, another new row will be added outside of the ProdCat group in the data region as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/total-sales-design.png)

Edit cell content

Modify the each total representation text as shown below.

1. Edit the third row, third column cell which contains Total text and set the following expression = "Total yearly sales of " & Fields!SubCat.Value in the expression builder.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/set-expression-for-subcat-field.png)

2. Edit the fourth row, second column cell which contains Total text and set the following expression = "Total yearly sales of " & Fields!ProdCat.Value in the expression builder.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/set-expression-for-prodcat-field.png)

3. Remove the Total text value from the fifth row, second column cell.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/remove-cell-content.png)

4. Edit the fifth row, first cell content as Grand Total. Now the table design will look like below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/edit-cell-content.png)

Now, the final design will look like below:

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/final-design-after-editing-cell-content.png)

Format total row

In the below design Background color of the total column is set with three different colors . This will differentiate the totals at details level, and product category Level. Set the font weight of the expression fields as Bold to improvise the table design. Also the background color of the tablix header row is changed to improvise the report presentation.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/format-total-row.png)

Design and preview

The final report design will look like below:

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/final-report-design.png)

On report preview, the total yearly sales of each product, product category and grand totals at details group level will be displayed like below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/total-sales-preview.png)

Navigate to the last page of the report in preview. The grand total details will be displayed at the end of the report like below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-add-grouping-and-totals/grand-total-design.png)

Conditional Formatting

Conditional formatting can be used to enhance the visual appearance of the tablix data region in the report design. In this section, basic report design to configure conditional formatting in a tablix data region is explained step by step.

Create dataset

To bind data to the tablix data region, create a dataset in your report. In this section, the following query is used to fetch customer details data from **Customers** table in **NorthWind** database.

```
`sql
```

```
Select CustomerID, CompanyName, Address, City, PostalCode, Country From Customers
```

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/dataset-list-view.png)

Create parameter

To apply conditional formatting based on report parameter, create a parameter in your report. Refer [Create Parameter](#) and [Define Available Value](#) section to create and assign value to the parameter. In this section, the **FormattingRow** parameter is created and assigned **Country** data field as query value. Refer the below snaps for better understanding.

New parameter creation

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/parameter-creation-panel.png)

Assign value

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/assign-available-value.png)

Parameter list view

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/parameter-list-view.png)

Add table data region

Design a simple table that contains following data from the fields in the dataset:

- CustomerID
- CompanyName
- City
- PostalCode
- Country

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/initial-design.png)

Refer [Simple Table Design](#) section to create above table design.

Formatting table

To improve the tablix design apply background color, font style to the table header by following the below steps:

1. Select the first row of table and click on the **Properties** icon in the configuration panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/set-cell-properties.png)

2. Set font color, font style and background properties for header row as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/set-cell-properties-output.png)

Refer [Properties Panel](#) section to set or edit values in properties panel.

Apply conditional formatting

To highlight the table rows which has the **Country** value selected in parameter drop-down at runtime, follow the below steps:

1. Select the **Details Row** group in the tablix. Now, the common properties for selected cells will be listed in the properties panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/select-details-group.png)

2. In the properties panel select **BackgroundColor** property and then click on the square icon in the right corner of the respective property.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/open-expression-menu-in-background-color-property.png)

3. Click on **Expression** menu to open the expression builder. Set the following conditional expression in the text area,
`=IIf(Fields!Country.Value=Parameters!FormattingRow.Value,"#e2f2bf","Transparent")`
and click on the **OK** button.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/set-condition-for-background-property.png)

Preview report

On report preview, select the country in the parameter drop-down and click on the **View Report** button. Now, based on the selected country the rows which contains the selected country name will be highlighted as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-conditional-formatting/conditional-fomatting-output.png)

Repeat Headers on Each Page in SSRS

To repeat the headers on every page of the report follow the below steps.

1. Refer [Simple Table Design](#) section and create a table report as shown below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-repeat-headers/simple-table-design.png)

2. Click on the surface of the table design to enable grouping panel.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-repeat-headers/enable-grouping-panel.png)

3. Refer [Advanced Mode](#) section to enable advanced mode in grouping panel.
4. Once you click on the **Advanced Mode**, it will show the static columns in both row and column group. Now, select the static column presented in row group pane (Header Row Group)

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-repeat-headers/select-static-group.png)

5. In the properties panel, enable **RepeatOnNewPage** property checkbox and set **KeepWithGroup** as **After**.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-repeat-headers/set-repeat-header-properties.png)

Now, preview the report and the header rows will now show up on every page.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-repeat-headers/report-preview-first-page.png)

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-repeat-headers/report-preview-last-page.png)

Matrix

Matrix can be used to display summarized data. It allows to group and summarize data by both rows and columns. A simple matrix design contains a row group, a column group, a corner cell, and a data cell.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/initial-matrix-design.png)

Properties

Refer the [Tablix Properties](#) section to set and edit properties value for matrix report item.

Add matrix to the report

1. The table report item is listed in the item panel under the **Data Regions** category.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/item-panel-view.png)

2. Drag and drop the table report item into the design area from the item panel.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-table.png)

Add row and column groups

1. Refer [Add parent group](#) section and add a parent row group to the table.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/add-row-group.png)

2. Now, select second column first cell and right click. In the cell menu, click on the **Parent Group** under the **Column Group** category. Refer [Parent column group](#) section and add a parent column group to the table.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/create-column-group.png)

3. When you add column group, a tablix corner cell will be create in the data region.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/corner-cell.png)

4. Delete the second row and add the label text in corner cell.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/add-label-in-corner-cell.png)

5. Now, delete the **Details** group from the table using the grouping panel.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-details-group.png)

In the delete confirmation dialog, choose **Delete group only** option and click on the **OK** button.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-group-confirmation.png)

6. Refer [Delete Columns](#) section and delete the excess columns in the right side of the column group.

Now, the matrix design with a row group and column group will be look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/basic-design.png)

Design SSRS Matrix Report

This section describes the steps to design yearly sales report of a company using SSRS matrix report item.

Create dataset

To present data in the matrix format, create a dataset and bind data to the matrix data region. In this designing section, the following dataset query is used for dataset creation.

```
'sql
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
FROM Production.ProductSubcategory PS INNER JOIN
Sales.SalesOrderHeader SOH INNER JOIN
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryId =
P.ProductSubcategoryId INNER JOIN
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryId
'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Add matrix to the report

1. The table report item is listed in the item panel under the **Data Regions** category.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/item-panel-view.png)

2. Drag and drop the table report item into the design area from the item panel.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-table.png)

3. Assign dataset to the **Dataset** property of table report item.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/matrix/assign-data.png)

Add parent row group

1. Select the tablix data region in the design area, now the **Grouping Panel** will be enabled in the design view.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/matrix/enable-grouping-panel.png)

2. To add a group, go to **Row Groups** pane in grouping panel and open the context menu on the **Details** group field.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/matrix/open-group-menu.png)

3. From the context menu, click on **Parent Group...** option under **Add Group** category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/matrix/select-parent-group.png)

4. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

5. Choose **Product Category** field in the **Group by** drop-down list and click on the **OK** button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/assign-field-for-parent-group.png)

Add parent column group

1. Now, select second column first cell and right click. In the cell menu, click on the **Parent Group** under the **Column Group** category.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/parent-column-group-menu.png)

- Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/create-column-group.png)

- Once you click on the Parent Group option, a Tablix Group dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

- Choose Order Year field in the Group by drop-down list and click on the OK button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/matrix/choose-column-parent-group.png)

Delete rows and columns

- Select the second row and right-click in the row gripper. Choose Delete Rows option.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-unused-row.png)

- Select the cells of last two column and right-click in the cell. Choose Delete Columns option.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-unused-column.png)

Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/add-parent-group-output.png)

Delete details group

- To delete a Details group , click on the icon in the right corner of the Details group member field in the grouping panel.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-details-group-menu.png)

- Click on Delete Group option in the menu. Now, the Delete Group confirmation dialog will be launched.
- Choose Delete group only option and click on the OK button.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-group-confirmation.png)

Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/delete-details-group-output.png)

Add child row group

1. In the Row Groups pane, open the group menu in ProdCat1 group field.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/open-group-menu-to-add-child-group.png)

2. From the context menu, click on Child Group... option under Add Group category.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/child-row-group.png)

3. Once you click on the Child Group option, a Tablix Group dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

4. Here, Sub Category field is chosen as child row group in tablix data region. Click on the OK button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/matrix/select-child-row-group.png)

Now, the matrix design will look like below.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/matrix/child-row-group-output.png)

Add child column group

1. In the Column Groups pane, open the group menu in OrderYear1 group field.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/open-group-menu-to-add-column-child-group.png)

2. From the context menu, click on Child Group... option under Add Group category.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/choose-child-group-option.png)

3. Once you click on the **Child Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

4. Here, **OrderQtr** field is chosen as child column group in tablix data region. Click on the **OK** button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/matrix/choose-child-group-field.png)

Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/add-child-column-group-output.png)

Calculate a summary

Now, a matrix report is created with row groups and column groups. To calculate the total sales amount for each product category, assign the **=Sum(Fields!Sales.Value)** expression to the last cell in third row of the matrix design as shown below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/calculate-fields.png)

To set expression in matrix cell, refer [Assign or edit expression into table cell](#) section.

Format data

By default, the summary data for the **Sales** field displays a general number. To format the **Sales** field to display the number as currency, set the format for **Sales** field cell using the **Format** property.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/format-data.png)

Change width or height

The matrix report item expands horizontally as well as vertically. You can resize rows and columns to display the data without wrapping. To resize rows and column in matrix data region, refer [Resize tablix data region](#). Here, the width of columns in matrix design is resized to minimum required width as shown below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/resize-output.png)

Merge matrix cells

After adding row and column groups, corner cells are created in the matrix design. You can use the corner cell to display any label content or data. The number of corner cell in a matrix design is based on the number of row and column groups in the matrix. In the above matrix design, it has four cells in its corner area. The cells are arranged in two rows and two columns. The four cells are not used in this report and it can be merged as one. To merge the corner cells, refer [Merge cells in corner area](#)

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/merge-corner-cell.png)

Format matrix design

In the below design background color and font styles are changed in matrix cells to improvise the report design.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/format-matrix-report.png)

Refer the [Cell Properties](#) to style the matrix cell.

Report preview

On report preview, the yearly sales report of a company will be displayed like below,

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/matrix/matrix-design-preview.png)

List

List report item can be used to create free-form layouts. You can arrange report items to create a form with text boxes, images, and other data regions placed anywhere within the list. It acts as a container to place multiple report items side by side to design a free-form layout. A simple list design has a single cell in a row associated with the detail group and the cell contains a **Rectangle** report item.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/initial-list-design.png)

By default, no group expression is defined on List. When no group expression is defined, List repeats for each row in the datasource. When a group expression is defined on a List, the List repeats for each group in the datasource.

Like tables and matrices, list report items are implemented as a Tablix data region.

Add report items in list

To add report items in list cell, drag report items from itempanel or data fields from the **Data** panel to the cell. By default, the cell contains a rectangle that acts as a container.

![Initial design](/static/assets/on-premise/images/report-designer/report-items/list/drag-and-drop-report-item-to-list.png)

Now, the report item will be dropped in the list container as shown in the below snap.

![Initial design](/static/assets/on-premise/images/report-designer/report-items/list/drag-and-drop-report-item-to-list-design.png)

When you right click in the list cell, the menu that appears is respect with the cell. If you insert a report item into the cell using the **Insert** option from cell context menu, the rectangle report item in the cell will be replaced by a new report item. So, to insert items inside the list cell, drag and drop the required report item into the cell containing rectangle.

In the following snap a simple design is created using the list to display product details.

![Initial design](/static/assets/on-premise/images/report-designer/report-items/list/initial-design.png)

On report preview, the List repeats for each row in the datasource.

![Initial design](/static/assets/on-premise/images/report-designer/report-items/list/list-reportitem-design-preview.png)

Properties

Refer the [Properties panel](#) section before proceeding with the below properties.

Data

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/data-category-property.png)

Dataset

This property is used to assign the dataset to the list. The available datasets in the report will be listed in the **Dataset** property dropdown. You can choose the desired dataset from the drop-down.

Each list report item can only show data from one dataset.

![Data category](/static/assets/on-premise/images/report-designer/report-items/tablix/dataset-drop-down-view.png)

Refer [Create Data](#) section to add dataset to your report.

Filter

Filters is used to filter the data in the list. To open the **Filter** dialog, click on the **Set Filters...** button.

Now, the filter dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/filters-dialog.png)

Refer [Filter Data](#) section to add/remove filters in the filter dialog.

Sort

To sort the numeric or string field in the list, sorting can be used. In list, the sorting can be applied to the whole data region or for each group, including the details group. To open the sort dialog, click on the **Set Sorts...** button. Now, the sort dialog will be opened like below.

![Sort dialog](/static/assets/on-premise/images/report-designer/report-items/tablix/sort-dialog.png)

Refer [Sort Data](#) section to add/remove sort expressions in the sort dialog.

Appearance

The border style, color, width and background color properties are used to style the list and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

Border

Border properties are used to add or customize the border around a list item to visually separate it in the report design. To set border properties to the list item using properties panel refer [Border Properties](#) section.

Background color

Using the background color property you can color the list background. To set background color using properties panel refer [Background color](#) section.

Position

Position property is used to set the width, height, left and top position of the list in the report design. To handle these properties using properties panel refer [Position](#) section.

Visibility

Visibility property is used to conditionally show or hide the list report item on report preview or export action. To set visibility of tablix item using properties panel refer [Visibility](#) section.

| | |
|-----------------------------------|----------------|
| Design ssrs rdl report using list | Set expression |
|-----------------------------------|----------------|

Miscellaneous

Custom Attributes

This property can be used to set the values for list report item custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for list report item using properties panel refer [Tooltip](#) section.

[Set expression](#)

An expression can be set to few properties of the list report item to process the property values based on expressions. To set expressions to the list report item properties, refer [Set Expression](#) section.

[Reset expression](#)

To Reset the expression applied to a property, refer [Reset Expression](#) section.

[Advanced properties](#)

Few properties of the list report item contains nested properties. To open and handle nested properties, refer [Advanced Properties](#) section.

[Design RDL report using list](#)

Refer [Design ssrs RDL report using list](#) section to learn how to design a simple list in your report.

[Design ssrs rdl report using list](#)

This section describes the steps to design **Mail Merge Report** using SSRS list report item.

[Create dataset](#)

To present data in the list format, create a dataset and bind data to the list data region. In this designing section, the following dataset query is used for dataset creation.

```
`sql
```

```
SELECT Emp.Photo as Photo, Emp.FirstName, Emp.LastName,
(DATENAME(WEEKDAY,Emp.BirthDate)+', '+DATENAME(DAY,Emp.BirthDate)+'
'+DATENAME(MONTH,Emp.BirthDate)+' '+DATENAME(YEAR,Emp.BirthDate)) as BirthDate,
Emp.Address, Emp.Title, Emp.Region, Emp.City, Emp.Country, Emp.TitleOfCourtesy, Rep.FirstName as
ReportingFirstName, Rep.LastName as ReportingLastName,
(DATENAME(WEEKDAY,Emp.HireDate)+', '+DATENAME(DAY,Emp.HireDate)+'
'+DATENAME(MONTH,Emp.HireDate)+' '+DATENAME(YEAR,Emp.HireDate))as HireDate,
Emp.Notes, Emp.HomePhone FROM Employees Emp INNER JOIN Employees Rep ON
Rep.EmployeeID=emp.ReportsTo
`
```

Refer [Create Data](#) section and create dataset using the above query. **Northwind** database is used here.

Configure a list

1. The list report item is listed in the item panel under the **Data Regions** category.

![List item listed in item panel](/static/assets/on-premise/images/report-designer/report-items/list/item-panel-view.png)

2. Drag and drop the list report item into the design area from the item panel.

![Drag and drop list report item into design area](/static/assets/on-premise/images/report-designer/report-items/list/drag-and-drop-list.png)

3. Respective list properties will be listed in the properties panel.

![List item with properties view](/static/assets/on-premise/images/report-designer/report-items/list/list-item-with-properties-view.png)

4. In the **DataSet** drop-down list choose the required dataset.

![Assign dataset](/static/assets/on-premise/images/report-designer/report-items/list/assign-dataset.png)

Initial design

Once you drop the list item, the **Table** renders with a single cell in a row associated with the detail group in the design area. By default, the cell contains a rectangle that acts as a container.

![List basic view](/static/assets/on-premise/images/report-designer/report-items/list/list-basic-view.png)

The cell in a list contains a rectangle. You can replace a default report item in a cell with any other report item.

Add report items

In list data regions, you can place the report items anywhere instead of being limited to a table layout. To create a mail merge template place the images, rectangles and textboxes inside of the list report item in a free-form manner.

1. Drag and drop two rectangle report items inside the list, to display the employee image and personal details side-by-side. Resize the list report item width and height to the required size.

![Add rectangle item](/static/assets/on-premise/images/report-designer/report-items/list/add-rectangle-item.png)

2. Then, drag and drop the image item in the left rectangle to display the employee image in the report.

![Add image item](/static/assets/on-premise/images/report-designer/report-items/list/add-image-item.png)

3. Bind the **Photo** field from the database to the image item.

![Bind data to the image](/static/assets/on-premise/images/report-designer/report-items/list/bind-data-to-the-image.png)

4. To display the employee name, add textbox in the bottom position of image report item.

![Add textbox item](/static/assets/on-premise/images/report-designer/report-items/list/add-textbox-item.png)

5. Now, right click in the textbox and click on **Expression** option.

![Assign expression in textbox](/static/assets/on-premise/images/report-designer/report-items/list/open-textbox-menu.png)

6. In the expression editor, choose **FirstName** field from the dataset.

![Bind data to the textbox](/static/assets/on-premise/images/report-designer/report-items/list/assign-dataset-field-in-textbox.png)

7. Similarly, assign **Lastname** field and create another expression. Then, customize the text appearance using the textbox properties in properties panel.

![Bind data field to the textbox](/static/assets/on-premise/images/report-designer/report-items/list/assign-expression-in-textbox.png)

Refer [Display dynamic text using expression](#) section to assign expression in textbox report item.

8. To display other employee details place other report items in the list item. In the below snap, the employee information are placed in the textboxes and the values are bound as expression to fetch dynamic data from the database.

![Mail merge template](/static/assets/on-premise/images/report-designer/report-items/list/mail-merge-template.png)

9. On report preview, each employee details will be displayed as list. To separate the each employee information you can drag and drop a line report item in the inside bottom position of the list cell.

![Mail merge template](/static/assets/on-premise/images/report-designer/report-items/list/add-line-report-item.png)

Report header

1. Enable the report **Header** to add a title to the report.

![Enable header tag](/static/assets/on-premise/images/report-designer/report-items/list/enable-header-tag.png)

Refer [Show or hide header and footer](#) section to add or remove header/footer in the report.

2. Now, add a rectangle report item in the report header area and a textbox within the rectangle.

![Add report items in header area](/static/assets/on-premise/images/report-designer/report-items/list/add-report-items-in-header-area.png)

3. Set the report title text in the textbox and customize the appearance of the title using the textbox and rectangle properties in properties panel as required.

![Report title text](/static/assets/on-premise/images/report-designer/report-items/list/report-title-text.png)

Final design

A final design of Mail Merge Report will look like below.

![Mail merge report design](/static/assets/on-premise/images/report-designer/report-items/list/final-design.png)

Report preview

On report preview, the mail merge report will be displayed like below,

![Mail merge report preview](/static/assets/on-premise/images/report-designer/report-items/list/report-preview.png)

Chart

Charts can be used to summarize the data in visual format. It helps to present data as bars, areas, lines and many other forms. To present the data, choose the appropriate chart type; it will determine how well the data can be interpreted while visualizing it in the chart form.

Add chart to the report

To add a **Chart** data region to the report, drag and drop the chart type of your choice from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/drag-chart-from-item-panel.png)

Now, the chart item will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/initial-view-of-column-chart.png)

Configure data and format the chart to visualize the data.

Here, Column chart type is used for demonstration.

Chart parts

The following snap shows the different elements used in the chart.

A placeholder image for chart parts.

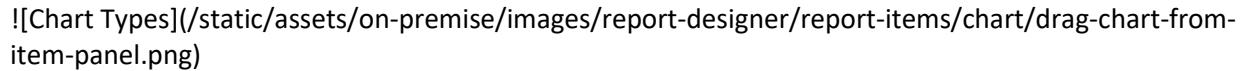
Column Chart

Column Chart allows you to compare values for a set of unordered items across categories through vertical bars ordered horizontally.

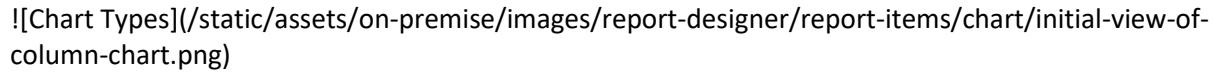
Add chart to the report

These types of charts are categorized under the **Comparison** category in item panel.

Drag and drop column chart from the item panel into design area.

A placeholder image for dragging a chart from the item panel.

Now, the column chart will be rendered in the design area and the chart properties will be listed in properties panel.

A placeholder image for the initial view of a column chart.

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

`sql

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
Sales.SalesOrderHeader SOH INNER JOIN  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
  
`
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Column Chart needs a minimum of 1 value element and 1 column element to showcase. The measure or expression field that you would like to analyze can be dropped into **Y Values** section. The dimension that you would like to categorize the measure, can be dropped onto **Columns** section. If you would like to categorize based on a series, then the respective dimension can be dropped onto **Rows** section in addition.

To configure data into column chart follow the steps:

1. To bind data to a chart report item placed in the design area, focus on that report item.
2. Click **Properties** icon in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/measures-dimensions-category.png)

5. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/y-value-chart-design-view.png)

6. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-menu.png)

7. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into Row(s) section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/row-field-design-preview.png)

Follow the above steps to assign data for other chart categories such as bar, line, area, radar and polar.

Format column chart

You can format the column chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format column chart follow the below steps:

1. Drag and drop the column chart into design area and resize it to required size.
2. Configure the data to the column chart.
3. Focus on the column chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

Name

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported chart types will be listed in the **Chart Type** property dropdown, you can switch to the required chart type based on the data.

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property.

The marker properties will be applied to the selected series in the **Choose Series** drop-down.

Show Data Label

Data label can be added to a chart series by using the **Show Data Label** property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property.

The data label properties will be applied to the selected series in the **Choose Series** drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable **Data Label** for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-design.png)

The series border properties will be applied to the selected series in the **Choose Series** drop-down.

You can also set properties based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-color-first-series.png)

Now, the selected color will be applied to the **Sales1** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/first-series-color-design.png)

Then, choose second series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-color-second-series.png)

Now, the selected color will be applied to the **Price1** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/second-series-color-design.png)

The series color properties will be applied to the selected series in the **Choose Series** drop-down.

You can also apply series color based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-design.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area-sketch.png)

These properties are listed under **Chart Area** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/chart-area-design.png)

Title

The chart title can be customized by editing the **Title Text** property of the chart.

To show/hide the chart title, toggle the **Show Chart Title** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Category Axis

Category axis displays the text labels instead of numbers. To use the categorical axis, toggle the **Enable Axis** checkbox under **Category Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Value Axis

Numeric axis uses numerical scale and displays numbers as labels. To use the categorical axis, toggle the **Enable Axis** checkbox under **Value Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/value-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Grid line

The Grid line properties can be set to category and value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/grid-line.png)

Category Axis

To show the grid line for category axis, enable the **Category Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/category-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/category-axis-minor-grid-lines.png)

Value Axis

To show the grid line for value axis, enable the **Value Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-axis-minor-grid-lines.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/report-preview-page.png)

Pie Chart

Pie Chart allows you to showcase proportionality of each item to the total in the form of pie-slices.

Add chart to the report

These types of charts are categorized under the **Proportion** category in item panel.

Drag and drop Pie chart from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/drag-chart-from-item-panel.png)

Now, the pie chart will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/initial-view-of-pie-chart.png)

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

```
'sql'
```

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
```

```
FROM Production.ProductSubcategory PS INNER JOIN
```

```
Sales.SalesOrderHeader SOH INNER JOIN
```

```
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
```

```
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryId =  
P.ProductSubcategoryId INNER JOIN
```

```
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID
```

```
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryID
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Pie Chart needs a minimum of 1 value element and 1 column element to showcase. The measure or expression field that you would like to analyze can be dropped into Y Values section. The dimension that you would like to categorize the measure, can be dropped onto Columns section. If you would like to categorize based on a series, then the respective dimension can be dropped onto Rows section in addition.

To configure data into pie chart follow the steps:

1. To bind data to a chart report item placed in the design area, focus on that report item.
2. Click Properties icon in the configuration panel, the property pane opens. Now, switch to DATA tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the Measures section; other type of columns and dimension expressions are listed under the Dimensions section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/measures-dimensions-category.png)

5. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the Measure section and drop it in the Y Values section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/y-value-chart-design-view.png)

6. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-menu.png)

7. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into **Row(s)** section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/row-field-design-preview.png)

Follow the above steps to assign data for other chart categories such as bar, line, area, pie and polar.

Format pie chart

You can format the pie chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format column chart follow the below steps:

1. Drag and drop the pie chart into design area and resize it to required size.
2. Configure the data to the pie chart.
3. Focus on the pie chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

<code class="language-text" style="word-break: break-word;">Name</code>

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported chart types will be listed in the **Chart Type** property dropdown, you can switch to the required chart type based on the data.

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property. The marker properties will be applied to the selected series in the **Choose Series** drop-down.

Show Data Label

Data label can be added to a chart series by using the [Show Data Label](#) property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property. The data label properties will be applied to the selected series in the [Choose Series](#) drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable [Data Label](#) for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/series-border-design.png)

The series border properties will be applied to the selected series in the [Choose Series](#) drop-down.

You can also set properties based on dynamic values, by using the [Expressions](#). Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the [Choose Series](#) drop-down and choose color in [Series Color](#) property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/series-color-first-series.png)

Now, the selected color will be applied to the [Sales](#) series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/first-series-color-design.png)

Then, choose second series in the [Choose Series](#) drop-down and choose color in [Series Color](#) property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/series-color-second-series.png)

Now, the selected color will be applied to the [OrderYear](#) series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/second-series-color-design.png)

The series color properties will be applied to the selected series in the **Choose Series** drop-down.

You can also apply series color based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/appearance-design.png)

Chart Area

You can customize the chart series color using the **Color Palatte** property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/color-palatte.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/color-palatte-design.png)

Title

The chart title can be customized by editing the **Title Text** property of the chart.

To show/hide the chart title, toggle the **Show Chart Title** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The **PageName** property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/pie-chart/report-preview-page.png)

Bubble Chart

Bubble Chart allows you to compare large number of data points represented as bubbles and showcase the difference through its size. The bubble chart requires two values per data point. By default, bubble chart display data points as circles.

Add chart to the report

These types of charts are categorized under the **Distribution** category in item panel.

Drag and drop bubble chart from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/drag-chart-from-item-panel.png)

Now, the bubble chart will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/initial-view-of-bubble-chart.png)

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

'sql

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
  
Sales.SalesOrderHeader SOH INNER JOIN  
  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN  
  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
\
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Bubble chart requires 3 fields (x, y and size) to plot a point. Here, size is used to specify the size of each bubble segment.

To configure data into bubble chart follow the below steps:

To bind data to a chart report item placed in the design area, focus on that report item.

Click **Properties** icon in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/chart-properties-pane.png)

The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/measures-dimensions-category.png)

Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/add-y-values-field.png)

You can add multiple numeric columns in the **Y-value** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/add-multiple-series.png)

Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/aggregation-settings-menu.png)

Expression field :

You can set the expression by which you want to visualize data in the bubble chart. Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

Then click on the **Expression** menu to open the expression editor.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/expression-menu.png)

In expression editor, provide the required expression and click **OK**.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/expression-editor.png)

Now, the label in **Y-Value** section for the respective field will be displayed with **<>Expr<>** tag.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/expression-indication.png)

Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/add-field-to-column-section.png)

Grouping:

You can group the added column element with another column, by adding the respective dimension element into **Row(s)** section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/add-field-to-rows-section.png)

Now, the chart design will look like below.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/final-design.png)

Size:

The Bubble size can be defined by adding a value into the Size section

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/add-field-to-xvalue-section.png)

Format bubble chart

You can format the bubble chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format bubble chart follow the below steps:

1. Drag and drop the bubble chart into design area and resize it to required size.
2. Configure the data to the bubble chart.
3. Focus on the bubble chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

Name

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property. The marker properties will be applied to the selected series in the [Choose Series](#) drop-down.

Show Data Label

Data label can be added to a chart series by using the [Show Data Label](#) property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property. The data label properties will be applied to the selected series in the [Choose Series](#) drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable [Data Label](#) for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-design.png)

The series border properties will be applied to the selected series in the [Choose Series](#) drop-down.

You can also set properties based on dynamic values, by using the [Expressions](#). Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the Choose Series drop-down and choose color in Series Color property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-color-first-series.png)

Now, the selected color will be applied to the Sales1 series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/first-series-color-design.png)

Then, choose second series in the Choose Series drop-down and choose color in Series Color property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-color-second-series.png)

Now, the selected color will be applied to the Price1 series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/second-series-color-design.png)

The series color properties will be applied to the selected series in the Choose Series drop-down.

You can also apply series color based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the Appearance category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/appearance-design.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/chart-area-sketch.png)

These properties are listed under Chart Area category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/chart-area-design.png)

Title

The chart title can be customized by editing the **Title Text** property of the chart.

To show/hide the chart title, toggle the **Show Chart Title** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Category Axis

Category axis displays the text labels instead of numbers. To use the categorical axis, toggle the **Enable Axis** checkbox under **Category Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Value Axis

Numeric axis uses numerical scale and displays numbers as labels. To use the categorical axis, toggle the **Enable Axis** checkbox under **Value Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/value-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Grid line

The Grid line properties can be set to category and value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/grid-line.png)

Category Axis

To show the grid line for category axis, enable the **Category Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/category-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/category-axis-minor-grid-lines.png)

Value Axis

To show the grid line for value axis, enable the **Value Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/value-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/value-axis-minor-grid-lines.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/bubble-chart/report-preview-page.png)

Scatter Chart

Scatter Chart allows you to compare large number of data points represented as dots irrespective of time.

Add chart to the report

These types of charts are categorized under the **Distribution** category in item panel.

Drag and drop scatter chart from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/drag-chart-from-item-panel.png)

Now, the scatter chart will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/initial-view-of-scatter-chart.png)

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

```
'sql
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
FROM Production.ProductSubcategory PS INNER JOIN
Sales.SalesOrderHeader SOH INNER JOIN
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =
P.ProductSubcategoryID INNER JOIN
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryID'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Scatter Chart needs a minimum of 1 value element and 1 column element to showcase. The measure or expression field that you would like to analyze can be dropped into **Y Values** section. The dimension that you would like to categorize the measure, can be dropped onto **Columns** section. If you would like to categorize based on a series, then the respective dimension can be dropped onto **Rows** section in addition.

To configure data into scatter chart follow the steps:

1. To bind data to a chart report item placed in the design area, focus on that report item.
2. Click **Properties** icon in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/measures-dimensions-category.png)

5. **Drag and Drop Measure Element:**

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/y-value-chart-design-view.png)

6. **Aggregate Options:**

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-menu.png)

7. **Drag and Drop Dimension Element:**

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into Row(s) section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/row-field-design-preview.png)

9. X-Value(s):

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **X Values** section to plot the data between two values to compare large number of data points.

Format scatter chart

You can format the scatter chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format scatter chart follow the below steps:

1. Drag and drop the scatter chart into design area and resize it to required size.
2. Configure the data to the scatter chart.
3. Focus on the scatter chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

Name

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported chart types will be listed in the **Chart Type** property dropdown, you can switch to the required chart type based on the data.

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property. The marker properties will be applied to the selected series in the **Choose Series** drop-down.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/marker-design.png)

Show Data Label

Data label can be added to a chart series by using the **Show Data Label** property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property. The data label properties will be applied to the selected series in the **Choose Series** drop-down.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/data-label.png)

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable **Data Label** for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/border-design.png)

The series border properties will be applied to the selected series in the **Choose Series** drop-down.

You can also set properties based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/series-color-first-series.png)

Now, the selected color will be applied to the **Sales1** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/first-series-color-design.png)

Then, choose second series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/series-color-second-series.png)

Now, the selected color will be applied to the **Price1** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/second-series-color-design.png)

You can also apply series color based on dynamic values, by using the [Expressions](#). Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

The series color properties will be applied to the selected series in the [Choose Series](#) drop-down.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the [Appearance](#) category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/appearance-design.png)

Chart Area

Chart Area properties such as border style, border width, border color, and background color can be used to customize the area of the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/chart-area-sketch.png)

These properties are listed under [Chart Area](#) category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

Title

The chart title can be customized by editing the [Title Text](#) property of the chart.

To show/hide the chart title, toggle the [Show Chart Title](#) checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Category Axis

Category axis displays the text labels instead of numbers. To use the categorical axis, toggle the [Enable Axis](#) checkbox under [Category Axis](#) category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Value Axis

Numeric axis uses numerical scale and displays numbers as labels. To use the categorical axis, toggle the [Enable Axis](#) checkbox under [Value Axis](#) category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/value-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Grid line

The Grid line properties can be set to category and value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/grid-line.png)

Category Axis

To show the grid line for category axis, enable the **Category Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/category-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/category-axis-minor-grid-lines.png)

Value Axis

To show the grid line for value axis, enable the **Value Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/value-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/value-axis-minor-grid-lines.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/scatter-chart/report-preview-page.png)

Line Chart

Line Chart allows you to showcase trends for analysis over a time period with data points connecting using straight lines.

Add chart to the report

These types of charts are categorized under the **Distribution** category in item panel.

Drag and drop column chart from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/drag-chart-from-item-panel.png)

Now, the line chart will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/initial-view-of-column-chart.png)

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

'sql

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
```

```
FROM Production.ProductSubcategory PS INNER JOIN  
Sales.SalesOrderHeader SOH INNER JOIN  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
,
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Column Chart needs a minimum of 1 value element and 1 column element to showcase. The measure or expression field that you would like to analyze can be dropped into **Y Values** section. The dimension that you would like to categorize the measure, can be dropped onto **Columns** section. If you would like to categorize based on a series, then the respective dimension can be dropped onto **Rows** section in addition.

To configure data into column chart follow the steps:

1. To bind data to a chart report item placed in the design area, focus on that report item.
2. Click **Properties** icon in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/measures-dimensions-category.png)

5. **Drag and Drop Measure Element:**

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/y-value-chart-design-view.png)

6. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-menu.png)

7. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into **Row(s)** section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/row-field-design-preview.png)

Follow the above steps to assign data for other chart categories such as bar, line, area, radar and polar.

Format line chart

You can format the line chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format line chart follow the below steps:

1. Drag and drop the line chart into design area and resize it to required size.
2. Configure the data to the line chart.
3. Focus on the line chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

Name

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported chart types will be listed in the **Chart Type** property dropdown, you can switch to the required chart type based on the data.

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property.

The marker properties will be applied to the selected series in the [Choose Series](#) drop-down.

Show Data Label

Data label can be added to a chart series by using the [Show Data Label](#) property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property.

The data label properties will be applied to the selected series in the [Choose Series](#) drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable [Data Label](#) for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

The series border properties will be applied to the selected series in the [Choose Series](#) drop-down.

You can also set properties based on dynamic values, by using the [Expressions](#). Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the [Choose Series](#) drop-down and choose color in [Series Color](#) property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/series-color-first-series.png)

Now, the selected color will be applied to the [Sales1](#) series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/first-series-color-design.png)

Then, choose second series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/series-color-second-series.png)

Now, the selected color will be applied to the **OrderYear** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/second-series-color-design.png)

The series color properties will be applied to the selected series in the **Choose Series** drop-down.

You can also apply series color based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/appearance-design.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design.

These properties are listed under **Chart Area** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/chart-area-design.png)

Title

The chart title can be customized by editing the **Title Text** property of the chart.

To show/hide the chart title, toggle the **Show Chart Title** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Category Axis

Category axis displays the text labels instead of numbers. To use the categorical axis, toggle the **Enable Axis** checkbox under **Category Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Value Axis

Numeric axis uses numerical scale and displays numbers as labels. To use the categorical axis, toggle the **Enable Axis** checkbox under **Value Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/value-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Grid line

The Grid line properties can be set to category and value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/grid-line.png)

Category Axis

To show the grid line for category axis, enable the **Category Axis** checkbox.

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-grid-line-advanced-properties.png)

Value Axis

To show the grid line for value axis, enable the **Value Axis** checkbox.

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-grid-line-advanced-properties.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/line-chart/report-preview-page.png)

Area Chart

Stacked Area Chart allows you to compare multiple measures through filled curves stacked one after the other vertically.

Add chart to the report

These types of charts are categorized under the **Distribution** category in item panel.

Drag and drop area chart from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/drag-chart-from-item-panel.png)

Now, the area chart will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/initial-view-of-area-chart.png)

Here, **Stacked Area** chart is used for demonstration.

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

```
'sql
```

```

SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales

FROM Production.ProductSubcategory PS INNER JOIN
Sales.SalesOrderHeader SOH INNER JOIN
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =
P.ProductSubcategoryID INNER JOIN
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID

WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')

GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryID
`
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Stacked Area Chart needs a minimum of 1 value element and 1 column element to showcase. The measure or expression field that you would like to analyze can be dropped into Y Values section. The dimension that you would like to categorize the measure, can be dropped onto Columns section. If you would like to categorize based on a series, then the respective dimension can be dropped onto Rows section in addition.

To configure data into area chart follow the steps:

1. To bind data to a chart report item placed in the design area, focus on that report item.
2. Click Properties icon in the configuration panel, the property pane opens. Now, switch to DATA tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the Measures section; other type of columns and dimension expressions are listed under the Dimensions section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/measures-dimensions-category.png)

5. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/y-value-chart-design-view.png)

6. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-menu.png)

7. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into **Row(s)** section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/row-field-design-preview.png)

Format area chart

You can format the area chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format area chart follow the below steps:

1. Drag and drop the area chart into design area and resize it to required size.
2. Configure the data to the area chart.
3. Focus on the area chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

Name

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported chart types will be listed in the **Chart Type** property dropdown, you can switch to the required chart type based on the data.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/sparkline/chart-type.png)

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/show-marker-design.png)

The marker properties will be applied to the selected series in the [Choose Series](#) drop-down.

Show Data Label

Data label can be added to a chart series by using the [Show Data Label](#) property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/data-label-design.png)

The data label properties will be applied to the selected series in the [Choose Series](#) drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable [Data Label](#) for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/border-design.png)

The series border properties will be applied to the selected series in the [Choose Series](#) drop-down.

You can also set properties based on dynamic values, by using the [Expressions](#). Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the Choose Series drop-down and choose color in Series Color property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/series-color-first-series.png)

Now, the selected color will be applied to the Sales series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/first-series-color-design.png)

Then, choose second series in the Choose Series drop-down and choose color in Series Color property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/series-color-second-series.png)

Now, the selected color will be applied to the Price1 series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/second-series-color-design.png)

The series color properties will be applied to the selected series in the Choose Series drop-down.

You can also apply series color based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the Appearance category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/appearance-design.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/chart-area-sketch.png)

These properties are listed under Chart Area category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/chart-area-design.png)

Title

The chart title can be customized by editing the **Title Text** property of the chart.

To show/hide the chart title, toggle the **Show Chart Title** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Category Axis

Category axis displays the text labels instead of numbers. To use the categorical axis, toggle the **Enable Axis** checkbox under **Category Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Value Axis

Numeric axis uses numerical scale and displays numbers as labels. To use the categorical axis, toggle the **Enable Axis** checkbox under **Value Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/value-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Grid line

The Grid line properties can be set to category and value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/grid-line.png)

Category Axis

To show the grid line for category axis, enable the **Category Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/category-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/category-axis-minor-grid-lines.png)

Value Axis

To show the grid line for value axis, enable the **Value Axis** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/value-axis-grid-line.png)

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-grid-line-advanced-properties.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/value-axis-minor-grid-lines.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/area-chart/report-preview-page.png)

Radar Chart

Radar chart allows you to display multivariate data in the form of a two-dimensional chart of three or more quantitative variables represented on axes starting from the same point.

Add chart to the report

These types of charts are categorized under the **Distribution** category in item panel.

Drag and drop column chart from the item panel into design area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/drag-chart-from-item-panel.png)

Now, the line chart will be rendered in the design area and the chart properties will be listed in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/initial-view-of-column-chart.png)

Create data

To present data in the chart, create a dataset and bind data to the chart data region. In this designing section, the following dataset query is used for dataset creation.

```
'sql
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
FROM Production.ProductSubcategory PS INNER JOIN
Sales.SalesOrderHeader SOH INNER JOIN
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =
P.ProductSubcategoryID INNER JOIN
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryID'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Column Chart needs a minimum of 1 value element and 1 column element to showcase. The measure or expression field that you would like to analyze can be dropped into **Y Values** section. The dimension that you would like to categorize the measure, can be dropped onto **Columns** section. If you would like to categorize based on a series, then the respective dimension can be dropped onto **Rows** section in addition.

To configure data into column chart follow the steps:

1. To bind data to a chart report item placed in the design area, focus on that report item.
2. Click **Properties** icon in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/chart/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/measures-dimensions-category.png)

5. **Drag and Drop Measure Element:**

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/y-value-chart-design-view.png)

6. **Aggregate Options:**

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/chart/aggregation-settings-menu.png)

7. **Drag and Drop Dimension Element:**

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into Row(s) section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/row-field-design-preview.png)

Follow the above steps to assign data for other chart categories such as bar, line, area, radar and polar.

Format line chart

You can format the line chart for better illustration of the view that you require, through the settings available in **Properties** tab.

To format line chart follow the below steps:

1. Drag and drop the line chart into design area and resize it to required size.
2. Configure the data to the line chart.
3. Focus on the line chart and click **Properties** icon in the configuration panel, the property pane opens.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/open-chart-properties.png)

You can see the list of properties available for the widget with default value.

General Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/name-property.png)

Name

Name property can be used to provide an unique name to the chart item in the report.

Basic Settings

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported chart types will be listed in the **Chart Type** property dropdown, you can switch to the required chart type based on the data.

Show Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also you can customize the legend text appearance. To set/reset legend properties, refer [Show Legend](#) property.

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

In the below snap, the chart has single series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/chart-with-single-series.png)

So, only one series is listed in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/single-series-list-in-drop-down.png)

If the chart has multiple series as below,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/chart-with-multiple-series.png)

Now, both series will be listed in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/multi-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/marker-design.png)

The marker properties will be applied to the selected series in the **Choose Series** drop-down.

Show Data Label

Data label can be added to a chart series by using the **Show Data Label** property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/data-label-design.png)

The data label properties will be applied to the selected series in the **Choose Series** drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset data label properties, refer [Enable Smart Label](#) property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

To apply smart label properties, enable **Data Label** for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

The series border properties will be applied to the selected series in the **Choose Series** drop-down.

You can also set properties based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose first series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/series-color-first-series.png)

Now, the selected color will be applied to the **Sales1** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/first-series-color-design.png)

Then, choose second series in the **Choose Series** drop-down and choose color in **Series Color** property palette.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/series-color-second-series.png)

Now, the selected color will be applied to the **OrderYear** series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/second-series-color-design.png)

The series color properties will be applied to the selected series in the **Choose Series** drop-down.

You can also apply series color based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/appearance-design.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design.

These properties are listed under **Chart Area** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/chart-area-design.png)

Title

The chart title can be customized by editing the **Title Text** property of the chart.

To show/hide the chart title, toggle the **Show Chart Title** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-title.png)

Using these properties the font color, font text, font style, border, background and position of the title can be customized in the chart design.

Category Axis

Category axis displays the text labels instead of numbers. To use the categorical axis, toggle the **Enable Axis** checkbox under **Category Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Value Axis

Numeric axis uses numerical scale and displays numbers as labels. To use the categorical axis, toggle the **Enable Axis** checkbox under **Value Axis** category in the chart properties.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/value-axis.png)

To set/reset axis properties, refer [Axis Properties](#) property.

Grid line

The Grid line properties can be set to category and value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/grid-line.png)

Category Axis

To show the grid line for category axis, enable the **Category Axis** checkbox.

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/category-grid-line-advanced-properties.png)

Value Axis

To show the grid line for value axis, enable the **Value Axis** checkbox.

You can also enable the **Minor Grid Lines** and customize the major and minor gridline style and color in the **Advanced Options** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/column-chart/value-grid-line-advanced-properties.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for chart report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for chart item using properties panel refer [Tooltip](#) section.

Preview report

1. To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Preview icon in design view](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/preview-icon.png)

2. Now, the report preview can be visualized like below.

![Chart report preview](/static/assets/on-premise/images/report-designer/report-items/chart/radar-chart/report-preview-page.png)

Chart Legend

A Legend is a text used to describe the data plotted. This allows you to toggle the visibility of legend in chart and also customize the legend appearance in chart design. This property is listed under the basic settings category in the chart properties panel.

Show or hide legend

To show/hide legend in the chart surface, toggle the **Show Legend** checkbox in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/show-legend-checkbox.png)

If you enable the **Show Legend** checkbox, the chart legend will be displayed in the bottom position of chart area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/chart-legend-indication.png)

You can also enable or disable the legend based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Format legend

You can customize the legend appearance through the properties provided in the **Advanced** menu of **show legend** property. Click on the square icon in the right side of the **Show Legend** checkbox and click on **Advanced** option in the menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/show-legend-advanced-menu.png)

Now, the legend properties will be displayed in the **Advanced Options** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/advanced-properties.png)

Show border

Show border property can be used to set border for legend area in the chart. To enable legend border, toggle the **Show Border** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/enable-legends-border.png)

Now, default border will be enabled to the legend area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/enable-border-design-view.png)

Also, set of border properties will get enabled in the **Advanced Options** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/legend-border.png)

Using the border color and width property, you can customize the border for the legend area. You can also set border properties based on dynamic values, by using the Expressions. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/expression-menu.png)

Background color

Background color property can be used to set the background color for the legend area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/background-color.png)

Now, the background color property will be applied to the legend area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/set-background-property.png)

You can also apply background color based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Font

The font family, font size, and font color properties can be used to customize the legend text.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/font-property.png)

You can also apply font properties based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/font-property-expression.png)

Font Style and Weight

The font style and weight properties can be used to set the style and weight for the legend text.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/font-style.png)

You can also apply font style and weight properties based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/font-style-expression.png)

Title

Title property can be used to add the title to the legend. Specify the required title text in the title property textbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/title-text.png)

Now, the provided title text will be updated in the chart legend area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/set-title-text.png)

You can also apply title text based on dynamic values, by using the [Expressions](#). Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Legend position

You can place the legend in different position inside the chart area. This property can be used to position the legend inside the chart area. Supported positioning patterns are listed in the [Legend Position](#) drop-down.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/legend-position.png)

By default the legend will be placed in [BottomCenter](#) position. In the below snap, the legend is positioned at [LeftTop](#) position inside the chart area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/legend-position-property.png)

Enable custom bounds

This property can be used to customize the size and position of the chart legend inside the chart area. To set/reset the size and position of legend, toggle the [Enable Custom Bounds](#) checkbox. Once you enable the checkbox, the position and size properties will be listed in the [Advanced Options](#) menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-legend/enable-custom-bounds.png)

Increase or decrease the width, height, left and top values as required, to position the legend in the chart area.

If custom bounds property is enabled, the [Legend position](#) property will have no effect.

Chart Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point.

Show or hide marker

To show/hide marker in the chart series, toggle the [Show Marker](#) checkbox in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/show-marker-checkbox.png)

If you enable the [Show Marker](#) checkbox, the marker will displayed in the respective series of chart. In the [Choose Series](#) drop-down [Sales1](#) series is selected, so the marker is enabled for that specific series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/chart-marker-indication.png)

You can also enable or disable the marker based on dynamic values, by using the [Expressions](#). Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Format marker

You can customize the marker appearance through the properties provided in the Advanced menu of show marker property. Click on the square icon in the right side of the Show Marker checkbox and click on Advanced option in the menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/show-marker-advanced-menu.png)

Now, the marker properties will be displayed in the Advanced Options category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/advanced-properties.png)

Border

Border property can be used to set the border color and width for the data marker in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/border-properties.png)

In the below snap, the border color and width is applied to the markers in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/border-property-design.png)

Color

Color property can be used to set the color for the marker.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/marker-color.png)

Now, the color property will be applied to the marker in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/marker-color-design.png)

You can also apply marker color based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Marker type

There are different shapes you can add to the chart marker by using the Marker Type property such as rectangle, circle, diamond etc. Choose the required shape in the drop-down,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/marker-types.png)

In the below design, Circle shape is applied to the marker.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/marker-type-design.png)

You can also apply marker shapes based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Size

Size property can be used to customize the size of the marker in the chart design.

| | |
|------------------|-------------------------|
| Chart Data Label | Show or hide data label |
|------------------|-------------------------|

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/size-property.png)

Increase or decrease the size value in the size property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-marker/size-property-design.png)

Chart Data Label

Data label can be added to a chart series by using the **Show Data Label** property. The labels appear at the top of the data point, by default.

Show or hide data label

To show/hide data label in the chart series, toggle the **Show Data Label** checkbox in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/show-data-label-checkbox.png)

If you enable the **Show Data Label** checkbox, the label will displayed in the respective series of chart. In the Choose Series drop-down **Sales1** series is selected, so the data label is enabled for that specific series in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/data-label-indication.png)

You can also enable or disable the data label based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Format data label

You can customize the data label appearance through the properties provided in the **Advanced** menu of show data label property. Click on the square icon in the right side of the **Show Data Label** checkbox and click on **Advanced** option in the menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/show-data-label-advanced-menu.png)

Now, the data label properties will be displayed in the **Advanced Options** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/advanced-properties.png)

Show border

Show border property can be used to set border for data labels in the chart. To enable data label border, toggle the **Show Border** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/enable-data-label-border.png)

Now, default border will be enabled to the data label.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/enable-border-design-view.png)

Also, set of border properties will get enabled in the **Advanced Options** category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/data-label-border.png)

Using the border color and width property, you can customize the border for the data label. You can also set border properties based on dynamic values, by using the Expressions. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/expression-menu.png)

Background color

Background color property can be used to set the background color for the data label.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/background-color.png)

Now, the background color property will be applied to the data label.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/set-background-property.png)

You can also apply background color based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Font

The font family, font size, and font color properties can be used to customize the labels.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/font-property.png)

You can also apply font properties based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/font-property-expression.png)

Font Style and Weight

The font style and weight properties can be used to set the style and weight for the legend text.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/font-style.png)

You can also apply font style and weight properties based on dynamic values, by using the Expressions. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/font-style-expression.png)

Position

You can position the label to the top, center or bottom position of the segment by using the Position property for the chart types such as column, bar, stacked bar, stacked column, 100% stacked bar and 100% stacked column.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/position-property.png)

Now, the data labels will be center positioned in the chart series.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/position-design-colum.png)

The label can be positioned inside or outside the perimeter of the series by using the label position option for the chart types such as Pie, Doughnut, Pyramid and Funnel. You can use the **CustomAttributes** property in the chart report item, to position the label for Pie, Doughnut, Pyramid and Funnel chart types,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/custom-attributes.png)

For Pie and Doughnut chart,

| Position | Value |
|----------|-----------------------|
| Outside | PieLabelStyle=Outside |
| Inside | PieLabelStyle=Inside |

For Pyramid and Funnel Chart,

| Position | Value |
|----------|---------------------------|
| Outside | PyramidLabelStyle=Outside |
| Inside | PyramidLabelStyle=Inside |

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/set-userdefined-text.png)

You can also set position property based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Label Rotation

Labels can be rotated by using the **Label Rotation** property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/lable-rotation.png)

Increase or decrease the value in the **Label Rotation** property field to set the rotation angle for the data label.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/lable-rotation-design.png)

You can also set label rotation property based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Format

Format property is used format data label values. To open the format dialog click on the icon highlighted in the below snap or you can type the format directly in the textbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/format-property.png)

To apply **Number**, **Currency**, **Date**, **Time**, **Scientific**, **Percentage** or **Custom** formats using format dialog follow the steps provided in [Format](#) section.

You can also set format property based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Label

You can use case-sensitive, chart-specific keywords to represent an item that exists in the chart. The following is a list of chart keywords.

These keywords are listed in the **Label** property drop-down,

| Chart Keyword | Description |
|---------------|---------------------------------------|
| ----- ----- | |
| #VALX | X value of the data point. |
| #VALY | Y value of the data point. |
| #VALY2 | Y value #2 of data point. |
| #VALY3 | Y value #3 of data point. |
| #VALY4 | Y value #4 of data point. |
| #VALY5 | Y value #5 of data point. |
| #VALY6 | Y value #6 of data point. |
| #AXISLABEL | Axis data point label. |
| #INDEX | Data point index. |
| #PERCENT | Percentage of the data point Y value. |
| #TOTAL | Total of all Y values in the series. |

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/chart-keywords-list.png)

For example, if you want to show the values in the data labels as percentages, you can use choose a keyword **#PERCENT** in the drop-down.

You can also set label property based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

If **Use Value as Label** checkbox is enabled, **Label** property will have no effect in the data label.

Use Value as Label

Toggle this checkbox to display actual values of series in the data label on report preview.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/show-data-label/use-value-as-label.png)

If this checkbox is enabled, **Label** property will have no effect in the data label.

You can enable or disable this property based on dynamic values, by using the Expressions. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Chart Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity.

Enable or disable smart label

To manage overlapping of labels on report preview, toggle the Enable Smart Label checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/enable-smart-label.png)

If you enable the Enable Smart Label checkbox, the label properties will be applied to the data label in respective series of chart.

You can also enable or disable the smart label based on dynamic values, by using the Expressions. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Format smart label

You can customize the data label appearance through the properties provided in the Advanced menu of smart label property. Click on the square icon in the right side of the Enable Smart Label checkbox and click on Advanced option in the menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/advanced-menu.png)

Now, the smart label properties will be displayed in the Advanced Options category.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/advanced-properties.png)

Label Style

The label styles are listed in the Label Style drop-down.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/label-style.png)

Set the label type based on the type of chart that are used to present the data.

Value

Based on the selected Label Style, the values will be listed in the Value drop-down.

PieLabelStyle

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/pie-label-style.png)

FunnelLabelStyle

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/funnel-label-style.png)

PyramidLabelStyle

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/pyramid-label-style.png)
BarLabelStyle
![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/bar-label-style.png)
LabelStyle

Common label styles will be listed in the **Value** field drop-down for **Label Style** type. The selected **Value** applied to the chart data region based on the selected chart type.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/default-label-style.png)

In the below design, pie chart is used of demonstration.

Select **PieLabelStyle** in **Label Style** drop-down and choose **Outside** in **Value** field. Now, the chart design will look like below.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/label-style-outside.png)

Now, choose **Inside** in **Value** field. Now, the chart design will look like below.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/smart-label/label-style-inside.png)

On above snap notice that some labels are shown outside for better design, the smart label property manages to avoid overlapping of labels and improves the chart design.

Chart Axis

Charts typically have two axes that are used to measure and categorize data: a vertical (y) **primaryYAxis**, and a horizontal (x) **primaryXAxis**. You can customize the axis appearance through the properties provided under the **Category Axis** and **Value Axis** category of chart properties. To customize category axis change the properties under the **Category Axis** property and for value axis change the properties under the **Value Axis** category.

Show or hide axis

To show/hide axis in the chart surface, toggle the **Enable Axis** checkbox in the properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/enable-axis-checkbox.png)

If you enable the **Enable Axis** checkbox, the chart axis will be displayed in the chart area.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/axes-types.png)

You can also enable or disable the chart axis based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Axis Title

The title property in the axis provides options to customize the text and font of the axis title. You can customize the font family, font style, font weight, alignment and size. Axis does not display the title, by default.

To display title text for axis, provide the text in **Axis Title** textbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/axis-title-property.png)

Now, the text will be displayed in the respective axis of chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/axis-title-design.png)

To set font weight, color, style, size and alignment, open the **Advanced Menu** of axis title property.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/title-advanced-property.png)

Now, the font properties will listed in the **Advanced Options** menu,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/title-advanced-properties.png)

You can also set the chart axis title text based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Line Style

Line style property can be used to set the line style, width, and color of the axis line.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/line-style.png)

In the below snap, line style, color and width properties are applied to chart axes

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/line-style-design.png)

Label Overflow Mode

Label Overflow Mode property can be used to handle the display mode of the overlapping labels.

Trim

This option trims the end of overlapping label in the axis.

Hide

This option hides the overlapping label in the axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-overflow-mode.png)

Label Rotation

Label rotation property can be used to define the rotation angle for the axis labels to display in the chart design.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-rotation.png)

Increase or decrease the rotation angle in the label rotation numeric textbox,

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-rotation-design.png)

Label Format

Label format property can be used to handle different formatting options like display type, denominations, decimal places, currency culture and negative value display format to the value axis labels.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-format.png)

Refer [Format Data](#) section to set different formats to the axis label. Here, the **Currency** format is applied to the **Sales** values in Value axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-format-design.png)

You can also set the label format based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Label Font

The font family, font size, and font color properties can be used to customize the label text.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-font.png)

You can also apply font properties based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/label-font-design.png)

Tick

Tick properties can be used to set style, width, color, and length of the axis tick, and to set the visibility of the major and minor tick marks.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/tick-properties.png)

Major Ticks

To enable major ticks in the chart axis, toggle the **Enable Major Ticks** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/enable-major-ticks.png)

Now, the major ticks will be enabled in the respective chart axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/major-ticks-design.png)

To customize the tick color, width and length, open the **Advanced...** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/major-tick-menu.png)

| | |
|-------------|------|
| Chart types | Tick |
|-------------|------|

Click on **Advanced...** option, now the properties will be listed in the **Advanced Option** pane.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/major-tick-advanced-properties.png)

Minor Ticks

To enable minor ticks in the chart axis, toggle the **Enable Minor Ticks** checkbox.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/enable-minor-ticks.png)

Now, the minor ticks will be enabled in the respective chart axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/minor-ticks-design.png)

To customize the tick color, width and length, open the **Advanced...** menu.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/minor-tick-menu.png)

Click on **Advanced...** option, now the properties will be listed in the **Advanced Option** pane.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/minor-tick-advanced-properties.png)

You can also enable or disable ticks based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Tick Position

Tick position property can be used to position the ticks outside or inside of the chart axis.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/tick-position.png)

By default, ticks will be positioned outside of the chart axis, in the below snap position is changed as **Inside**.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-axis/tick-position-design.png)

You can also apply tick position based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Axis properties are not applicable for **Proportion** chart types.

Chart types

The supported chart types are listed in the itempanel under the following categories:

- Comparison
- Proportion
- Distribution

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/chart/chart-types-in-item-panel.png)

Choose an appropriate chart type, based on the type of data you are presenting.

Comparison

Column

| Types | Description |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Column | Column Chart allows you to compare values for a set of unordered items across categories through vertical bars ordered horizontally. |
| StackedColumn | Stacked Column Chart allows you to compare multiple measures through bars stacked one after the other vertically. |
| StackedColumnPercent | 100% Stacked Column Chart allows you to compare multiple measures through bars stacked one after the other vertically. |

Bar

| Types | Description |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Bar | Bar Chart allows you to compare values for a set of unordered items across categories through horizontal bars ordered vertically. |
| StackedBar | Stacked Bar Chart allows you to compare multiple measures through bars stacked one after the other horizontally. |
| StackedBarPercent | 100% Stacked Bar Chart allows you to compare multiple measures through bars stacked one after the other horizontally. |

Proportion

Pie

| Types | Description |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Pie | Pie Chart allows you to showcase proportionality of each item to the total in the form of pie-slices. |
| ExplodedPie | ExplodedPie Chart allows you to separate one or more sectors from the rest of the disk. This effect is used to either highlight a sector, or to highlight smaller segments of the chart with small proportions. |
| Doughnut | Doughnut Chart allows you to showcase proportionality of each item to the total in the form of doughnut-slices. To plot a doughnut chart, a minimum requirement of 1 value and 1 column is needed. |

Shape

| Types | Description |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Pyramid | Pyramid Chart allows you to make proportional comparison between values showcased as progressively increasing manner. To plot a pyramid chart, a minimum requirement of 1 value and 1 column is needed. |

| Chart types | Distribution |
|-------------|--------------|
|-------------|--------------|

| Funnel |Funnel Chart shows values across multiple stages in a process by highlighting different stages with different colors. It allows you make proportional comparison among values showcased in progressively decreasing manner.|

[Distribution](#)

[Area](#)

| Types | Description |
|-------|-------------|
|-------|-------------|

|-----|-----|

| Area |Area Chart allows you to compare values for a set of unordered items across categories through filled curves ordered vertically.|

| SmoothArea |An area chart where the data points are connected by a smooth line instead of a regular line. Compares the distribution of values over a time period connected using the smooth lines.|

| StackedArea |Stacked Area Chart allows you to compare multiple measures through filled curves stacked one after the other vertically.|

| StackedAreaPercent |100% Stacked Area Chart allows you to compare multiple measures through filled curves stacked one after the other vertically.|

[Line](#)

| Types | Description |
|-------|-------------|
|-------|-------------|

|-----|-----|

| Line |Line Chart allows you to showcase trends for analysis over a time period with data points connecting using straight lines.|

| SmoothLine |SmoothLine Chart allows you to showcase trends for analysis over a time period with data points connected using splines.|

| SteppedLine |SteppedLine chart allows you to compare the distribution of values over a time period connected using the stepped lines.|

| LineWithMarkers |LineWithMarkers chart allows you to compare changes over the same period of time for more than one group.|

| SmoothLineWithMarkers |Plotted values are represented with a marker point and those points are connected using a smooth line.|

[Scatter](#)

| Types | Description |
|-------|-------------|
|-------|-------------|

|-----|-----|

| Bubble |Bubble Chart allows you to compare large number of data points represented as bubbles and showcase the difference through its size.|

| Scatter |Scatter Chart allows you to compare large number of data points represented as dots irrespective of time.|

[Polar](#)

| Types | Description |
|-------|-------------|
|-------|-------------|

|-----|-----|

| | |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Radar | Radar chart allows you to display multivariate data in the form of a two-dimensional chart of three or more quantitative variables represented on axes starting from the same point. |
| Polar | Displays a series as a set of points that are grouped by category on a 360-degree circle. |

Chart Nested Data Regions

The data region such as a chart can be placed inside the another data region such as a tablix, to link more than one data region to the same dataset. This will provide different views of the same data. In the below steps, sales of each sub-category products for each product category group and order year is designed using chart and matrix data region.

Create dataset

To present data in the matrix format, create a dataset and bind data to the matrix data region. In this designing section, the following dataset query is used for dataset creation.

```
`sql
```

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
Sales.SalesOrderHeader SOH INNER JOIN  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
  
WHERE (SOH.OrderDate BETWEEN '1/1/2002' AND '12/31/2003')  
  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
`
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Add table to the report

1. The table report item is listed in the item panel under the **Data Regions** category.

![Tablix listed in item panel](/static/assets/on-premise/images/report-designer/report-items/tablix/item-panel-view.png)

2. Drag and drop the table report item into the design area from the item panel.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/tablix/drag-and-drop-table.png)

3. Assign dataset to the **Dataset** property of table report item.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/assign-data.png)

Add parent row group

1. Select the tablix data region in the design area, now the **Grouping Panel** will be enabled in the design view.

![Enable grouping panel](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/enable-grouping-panel.png)

2. To add a group, go to **Row Groups** pane in grouping panel and open the context menu on the **Details** group field.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/open-group-menu.png)

3. From the context menu, click on **Parent Group...** option under **Add Group** category.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/select-parent-group.png)

4. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

5. Choose **Product Category** field in the **Group by** drop-down list and click on the **OK** button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/assign-field-for-parent-group.png)

Add parent column group

1. Now, select second column first cell and right click. In the cell menu, click on the **Parent Group** under the **Column Group** category.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/parent-column-group-menu.png)

2. Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/create-column-group.png)

3. Once you click on the **Parent Group** option, a **Tablix Group** dialog will be opened to configure the grouping.

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/tablix-insert-or-delete-group/tablix-group-dialog.png)

4. Choose **Order Year** field in the **Group by** drop-down list and click on the **OK** button

![Open group menu](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/choose-column-parent-group.png)

Delete rows and columns

1. Select the second row and right-click in the row gripper. Choose **Delete Rows** option.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/delete-unused-row.png)

2. Select the cells of last two column and right-click in the cell. Choose **Delete Columns** option.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/delete-unused-column.png)

Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/add-parent-group-output.png)

Delete details group

1. To delete a **Details** group , click on the icon in the right corner of the **Details** group member field in the grouping panel.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/delete-details-group-menu.png)

2. Click on **Delete Group** option in the menu. Now, the **Delete Group** confirmation dialog will be launched.
3. Choose **Delete group only** option and click on the **OK** button.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/delete-group-confirmation.png)

Now, the matrix design will look like below.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/delete-details-group-output.png)

Add chart into the matrix

1. Drag and drop a rectangle and place inside the second row, last cell.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/add-rectangle.png)

Use rectangles to control the sizing of inserted chart item.

2. Now, drag a Exploded-pie chart from the item panel and place inside the rectangle.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/add-chart-into-cell.png)

3. The chart will be placed inside rectangle within table cell as below,

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/add-chart-design.png)

4. Resize the chart and matrix design as required.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/resize-table-design.png)

Assign data to chart

Select the chart report item in table cell. Now, the chart properties will be listed in the properties panel. Then, click on the DATA tab.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/open-chart-properties.png)

Based on the dataset assigned for parent matrix report item, the dataset field will be listed in the DATA assign panel. The numeric columns and numeric expressions are listed under the Measures section; other type of columns and dimension expressions are listed under the Dimensions section.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/data-assign-panel.png)

Now, drag and drop Sales field from Measures section and drop it in the Y Values section.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/add-y-value.png)

Then, drag and drop SubCat field from the Dimensions section to measure against any of the selected numeric column(s) in Y Value(s) section, and drop into the Column(s) section.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/add-column-field.png)

Format matrix

In the below design background color and font styles are changed in matrix cells to improvise the report design.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/format-matrix-report.png)

Refer the [Cell Properties](#) to style the matrix cell.

Format chart

Select the chart in the table cell. Now, the chart properties will be listed in the properties panel.

Enable the chart data label checkbox,

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/enable-data-label.png)

Disable the show legend checkbox,

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/show-legend.png)

Disable the chart title property,

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/chart-title.png)

Now, set the data label text value as `#AXISLABEL` in the `Label` property.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/data-label-value.png)

Refer [Show Data Label](#) section to customize data label in chart.

To set label position for pie chart, set `PieLabelStyle=Outside` as value in `UserDefined` property.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/set-label-position.png)

Set the series color using the `Color Palette` property.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/color-palette.png)

You can further customize the appearance of the chart using the chart properties in properties panel.

Final design

Resize the tablix data region to the required size in the design area. Now, the final design will look like below,

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/final-design.png)

Report preview

To see the report preview, click on the **Preview** button in the top-right corner of the report header.

![Matrix simple design](/static/assets/on-premise/images/report-designer/report-items/chart/nested-data-region/final-design-preview.png)

Add filters to chart data region

Filters can be used to filter data in the chart data region to include or exclude specific values from display or to provide a different view of the dataset in multiple data regions. When processing the report, the filters applied in the report parts are processed first on the dataset, and then on the data region, and then on groups.

Set filter on a chart category group

1. Select chart data region in the design area.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/select-data-region.png)

2. Switch to **DATA** tab,

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/switch-to-data-tab.png)

3. In the **DATA** assign panel, under the column section click on the **Settings** icon.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/filter-data-menu.png)

4. Now, click on the **Filters...** option in the menu. Now, the filter dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/filters-dialog.png)

5. Refer [Filter Data](#) section and create required filter expression in the filter dialog and click **OK**.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/create-filter-expressions.png)

In the above image, the filter equation is created to filter the data in the data region based on **Product Category** data field.

On report preview, using the **OrderYear** and **ProductCategory** report parameters, the sales of **Q1**, **Q2**, **Q3** and **Q4** is displayed in the chart data region based on applied filters.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/report-preview.png)

Similarly, you can set filters on a chart series group.

Sort data in a chart data region

You can control the order in which data appears in a chart data region by representing sort expressions. You can sort the data either in ascending or descending order.

| | |
|-----------|-----------------------------------------------|
| Indicator | Set sort expression on a chart category group |
|-----------|-----------------------------------------------|

Set sort expression on a chart category group

1. Select chart data region in the design area.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/sort-data-in-chart-data-region/select-data-region.png)

2. Switch to **DATA** tab,

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/switch-to-data-tab.png)

3. In the **DATA** assign panel, under the column section click on the **Settings** icon.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/add-filter-to-chart-data-region/filter-data-menu.png)

4. Now, click on the **Sorts...** option in the menu. Now, the sort dialog will be opened like below.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/sort-data-in-chart-data-region/sort-dialog.png)

5. Refer [Sort Data](#) section to and create required sort expression in the sort dialog and click on the **OK** button.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/sort-data-in-chart-data-region/new-sort-expression.png)

A sort expression is created to sort **ProductCategory** fields in descending order (i.e Z-A). On report preview, the **ProductCategory** field values will be sorted in Z-A alphabetical order.

![Filter dialog](/static/assets/on-premise/images/report-designer/report-items/chart/sort-data-in-chart-data-region/sort-data-preview.png)

Similarly, you can set sort expression on a chart series group.

Indicator

Indicator report items are minimal representation of gauge used to visualize the state of a data. It is mostly used in **Tablix** to visually display the state of the cell data.

Add an indicator to the report

1. The indicator report item is listed in the item panel under the **KPI** category.

![Indicator item in item panel](/static/assets/on-premise/images/report-designer/report-items/indicator/item-panel.png)

2. Drag and drop the indicator report item into the design area from the item panel.

![Drag and drop indicator report item into design area](/static/assets/on-premise/images/report-designer/report-items/indicator/item-drag.png)

3. After dropping the indicator item in the design area, the respective item properties will be listed in the properties panel.

![Indicator item with properties view](/static/assets/on-premise/images/report-designer/report-items/indicator/designer-area.png)

Properties

Refer to the [Properties panel](#) section before proceeding with the following properties.

Basic settings

The border style, color, width, and background color properties are used to style the indicator and customize its appearance in the report design. These properties are listed under the [Basic settings](#) category in the properties panel.

Border

Border properties are used to add or customize the border around an indicator item to visually separate items in the report design. To set border properties to the indicator item using properties panel, refer to the [Border Properties](#) section.

Background color

Using the background color property, you can customize the indicator background color. To set background color using properties panel, refer to the [Background color](#) section.

Position

The position property is used to set the width, height, left and top position of the indicator in the report design. To handle these properties using properties panel, refer to the [Position](#) section.

Data

Dataset

This property is used to assign the dataset to the indicator report item. The available datasets in the report will be listed in the [Dataset](#) property dropdown. You can choose the desired dataset from the drop-down.

![Data category](/static/assets/on-premise/images/report-designer/report-items/indicator/dataset.png)

Refer to the [Create Data](#) section to add dataset to your report.

Indicator value

This property is used to assign value for indicator states. You can also set the indicator value based on dynamic values using expressions.

![Indicator Numeric Value](/static/assets/on-premise/images/report-designer/report-items/indicator/numeric-value.png)

Measurement unit

You can select Percentage or Numeric from the dropdown for indicator value measurement. If You select Percentage option, Minimum and Maximum properties will be visible in the property panel.

![Indicator Percentage Value](/static/assets/on-premise/images/report-designer/report-items/indicator/percentage-value.png)

| | |
|-----------|------------|
| Indicator | Properties |
|-----------|------------|

Minimum

The minimum property is used to assign minimum value to the indicator. You can also set the minimum property value based on dynamic values using expressions.

Maximum

The maximum property is used to assign maximum value to the indicator. You can also set the maximum property value based on dynamic values using expressions.

Indicator types

This property is used to select different types of predefined indicator states from the dropdown.

![Indicator Type](/static/assets/on-premise/images/report-designer/report-items/indicator/types.png)

Indicator states

This property is used to customize the states of indicator.

![Indicator States](/static/assets/on-premise/images/report-designer/report-items/indicator/states.png)

Customizing a state

- **Customizing an icon** - You can choose different icon for each state from the dropdown under **Icon** column.
- **Customizing a color** - You can choose different color for each state from the color picker under **Color** column.
- **Customizing start value** - You can assign start value for each state from the numeric textbox under **Start** column and also set the start value based on dynamic values using expressions.
- **Customizing end value** - You can assign end value for each state from the numeric textbox under **End** column and also set the end value based on dynamic values using expressions.

Adding a new state

You can add a new state by clicking the **Add** icon. The new state will be created with the following values

- **Icon** - Circle
- **color** - No Color
- **Start** - "
- **End** - "

![Adding indicator states](/static/assets/on-premise/images/report-designer/report-items/indicator/states-add.png)

Deleting a state

You can delete a state by clicking the **delete** icon.

![Deleting indicator states](/static/assets/on-premise/images/report-designer/report-items/indicator/states-delete.png)

Visibility

The visibility property is used to conditionally show or hide the indicator report item on report preview or export action. To set visibility of indicator item using properties panel, refer to the [Visibility](#) section.

| | |
|-------|----------------|
| Gauge | Set expression |
|-------|----------------|

Miscellaneous

Custom Attributes

This property can be used to set the values for indicator custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for indicator report item using properties panel refer [Tooltip](#) section.

Set expression

An expression can be set to few properties of the indicator report item to process the property values based on expressions. To set expressions to the indicator report item properties, refer to the [Set Expression](#) section.

Reset expression

To Reset the expression applied to a property, refer to the [Reset Expression](#) section.

Advanced properties

A few properties of the indicator report contain nested properties. To open and handle the nested properties, refer to the [Advanced Properties](#) section.

Gauge

Gauge report items are used to visualize the key performance indicators (KPIs) that track the amount of progress made towards achieving the targets or goals. When placed inside a table or matrix, it illustrates values inside every cell.

Gauge types

There are two types of gauges, namely **Linear** and **Radial**. These are listed in the item panel under the **Deviation** category.

![Gauge Types](/static/assets/on-premise/images/report-designer/report-items/gauge/gauge-types-in-item-panel.png)

See Also

- [Linear Gauge](#)
- [Radial Gauge](#)

Linear gauge

Linear gauges are rectangular in shape and oriented horizontally or vertically. These can be placed inside the **Tablix** and **Matrix**.

Add a linear gauge to the report

1. The linear gauge report item is listed in the item panel under the **Deviation** category.

![Linear gauge item in item panel](/static/assets/on-premise/images/report-designer/report-items/gauge/linear-gauge/item-panel.png)

2. Drag the linear gauge report item into the design area from the item panel.

![Drag and drop linear gauge report item into design area](/static/assets/on-premise/images/report-designer/report-items/gauge/linear-gauge/item-drag.png)

3. After dropping the linear gauge item in the design area, the respective item properties will be listed in the properties panel.

![Linear gauge item with properties view](/static/assets/on-premise/images/report-designer/report-items/gauge/designer-area.png)

Properties

Refer to the [Properties panel](#) section before proceeding with the following properties:

Basic settings

The border style, color, width, and background color properties are used to customize the appearance of a linear gauge in the report design. These properties are listed under the **Basic settings** category in the properties panel.

Border

Border properties are used to add or customize the border around a linear gauge item to visually separate items in the report design. To set border properties to a linear gauge item using properties panel, refer to the [Border Properties](#) section.

Background color

Using the background color property, you can customize the background color of a linear gauge. To set the background color using properties panel, refer to the [Background color](#) section.

Data

Dataset

This property is used to assign the dataset to a linear gauge report item. The available datasets in the report will be listed in the **Dataset** property dropdown. You can choose the desired dataset from the dropdown.

![Data category](/static/assets/on-premise/images/report-designer/report-items/gauge/linear-gauge/dataset.png)

Refer to the [Create Data](#) section to add dataset to your report.

Type

This property is used to select the type of linear gauge report item. You can select **Horizontal**, **Vertical**, or **Auto** from the dropdown.

![Linear Gauge Type](/static/assets/on-premise/images/report-designer/report-items/gauge/linear-gauge/type.png)

- **Horizontal**: Displays the linear gauge horizontally.
- **Vertical**: Displays the linear gauge vertically.
- **Auto**: Displays the linear gauge according to the width and height of the linear gauge.

Value

This property is used to assign the pointer value of a linear gauge. You can also set the pointer value based on dynamic values using [expressions](#).

Range

This property is used to assign minimum and maximum values of a linear gauge. You can also set the value for minimum and maximum properties based on dynamic values using [expressions](#).

Interval

This property is used to assign interval value between the [scale labels](#). You can also set the interval property value based on dynamic values using [expressions](#).

Pointer

Using this property, the pointer placement, type, marker style, marker length, width, and color can be customized in the linear gauge design.

Enable pointer

To show or hide the pointer in linear gauge, toggle the Enable Pointer checkbox.

Placement

Pointer placements can be Inside, Outside, or Cross. You can select any placement from the dropdown.

Type

You can select the pointer type, Marker or Bar, from the dropdown. If you select the Marker option, Marker Style and Marker Length properties will be visible in the property panel.

Marker style

You can select a marker style from the dropdown. The different styles are Circle, Rectangle, Triangle, or Diamond.

The following diagram represents the linear gauge with the Marker pointer type, Diamond marker style, and Cross pointer placement.

![Pointer Type](/static/assets/on-premise/images/report-designer/report-items/gauge/linear-gauge/pointer-property.png)

Marker length

This property is used to assign the pointer marker length value. You can also set the marker length property value based on dynamic values using [expressions](#).

Width

This property is used to assign the pointer width value. You can also set the width property value based on dynamic values using [expressions](#).

Color

Using the color property, you can customize the pointer color of a linear gauge.

Scale

Width

This property is used to assign the scale width value. You can also set the width property value based on dynamic values using [expressions](#).

Color

Using the color property, you can customize the scale color of a linear gauge.

Reverse direction

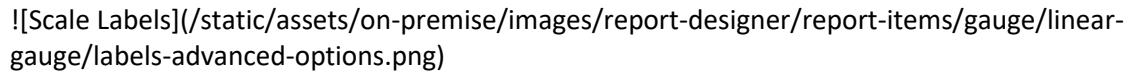
By clicking the Reverse Direction checkbox you can reverse the scale direction from maximum to minimum.



Labels

To show or hide the scale labels in a linear gauge, toggle the Label checkbox.

You can customize the scale label in the Advanced Options menu. To open and handle the nested properties of labels, refer to the [Advanced Properties](#) section.



Using these properties, the scale label font color, font family, font size, and font style can be customized in the linear gauge design.

Show labels at the end

To show or hide the scale labels at the end of the scale, toggle the Show Labels At End checkbox.

Placement

Scale label placement can be Inside, Outside, or Cross. You can select an option from the dropdown.

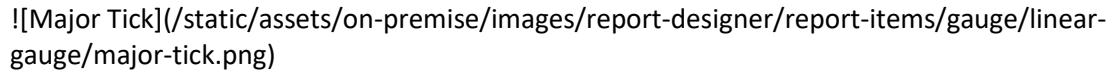
Tick mark

This property is used to customize major and minor ticks of a linear gauge.

Major tick and minor tick

To show or hide major and minor ticks in a scale, toggle the Major Tick and Minor Tick checkbox.

You can customize major and minor ticks in the Advanced Options menu. To open and handle the nested properties of major and minor ticks, refer to the [Advanced Properties](#) section.



Using this property, you can customize the major and minor ticks interval, color, length, and width.

Major tick placement and minor tick placement

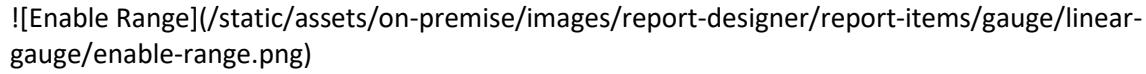
Major tick and minor tick placement can be Inside, Outside, or Cross. The placement can be selected from the dropdown.

Range

Using this property, the range placement, start value, end value, start width, end width, and color can be customized in the linear gauge design.

Enable range

To show or hide the range in a linear gauge, toggle the Enable Range checkbox.



Placement

Range placement can be Inside, Outside, or Cross. You can select any option from the dropdown.

Range

This property is used to assign the range **start** and **end** values of the linear gauge. You can also set the start and end properties value based on dynamic values using [expressions](#).

Width

This property is used to assign the range **start width** and **end width** values of the linear gauge. You can also set the start and end width properties value based on dynamic values using [expressions](#).

Color

Using the color property, you can customize the **range color** of a linear gauge.

Position

The position property is used to set the width, height, left, and top positions of a linear gauge in the report design. To handle these properties using properties panel, refer to the [Position](#) section.

Visibility

The visibility property is used to conditionally show or hide the linear gauge report item on report preview or export action. To set visibility of a linear gauge item using properties panel, refer to the [Visibility](#) section.

Miscellaneous

Custom attributes

This property can be used to set the values for linear gauge custom properties. To assign values for custom properties using properties panel, refer to the [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for gauge report item using properties panel refer [Tooltip](#) section.

Radial gauge

The radial gauges are degrees of circular to represent key performance indication. These can be placed inside the **Tablix** and **Matrix**.

Add a radial gauge to the report

1. The radial gauge report item is listed in the item panel under the **Deviation** category.

![Radial gauge item in item panel](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/item-panel.png)

2. Drag the radial gauge report item into the design area from the item panel.

![Drag the radial gauge report item into design area](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/item-drag.png)

3. After dropping the radial gauge item in the design area, the respective item properties will be listed in the properties panel.

![Radial gauge item with properties view](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/designer-area.png)

Properties

Refer to the [Properties panel](#) section before proceeding with the following properties:

Basic settings

The border style, color, width, and background color properties are used to customize the appearance of a radial gauge in the report design. These properties are listed under the **Basic settings** category in the properties panel.

Border

The border properties are used to add or customize the border around a radial gauge item to visually separate items in the report design. To set border properties to the radial gauge item using properties panel, refer to the [Border Properties](#) section.

Background color

Using the background color property, you can customize background color of a radial gauge. To set the background color using properties panel, refer to the [Background color](#) section.

Data

Dataset

This property is used to assign the dataset to a radial gauge report item. The available datasets in the report will be listed in the **Dataset** property dropdown. You can choose the desired dataset from the dropdown.

![Data category](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/dataset.png)

Refer to the [Create Data](#) section to add dataset to your report.

Type

This property is used to select the type of a radial gauge report item. You can select **Radial**, **Half Circle**, or **Auto** from the dropdown.

![Radial Gauge Type](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/type.png)

- **Radial**: Displays the radial gauge of start-angle(0) and sweep-angle(320).
- **Half Circle**: Displays the radial gauge of start-angle(90) and sweep-angle(180).
- **Auto**: Displays the radial gauge with custom start-angle and sweep-angle.

Value

This property is used to assign the pointer value of a radial gauge. You can also set the pointer value based on dynamic values using [expressions](#).

Range

This property is used to assign **minimum** and **maximum** values of a radial gauge. You can also set the values for minimum and maximum properties based on dynamic values using [expressions](#).

Interval

This property is used to assign **interval** value between the [scale labels](#). You can also set the interval property value based on dynamic values using [expressions](#).

Pointer

Using this property, the pointer placement, type, marker style, marker length, cap, width and color can be customized in the radial gauge design.

Enable pointer

To show or hide the pointer in radial gauge, toggle the **Enable Pointer** checkbox.

Placement

Pointer placements can be **Inside**, **Outside**, or **Cross**. You can select any placement option from the dropdown.

Type

You can select the pointer type as **Needle**, **Marker** or **Bar** from the dropdown. If You select **Marker** option, **Marker Style**, and **Marker Length** properties will be visible in the property panel.

Marker style

You can select a marker style from the dropdown. The different styles are **Circle**, **Rectangle**, **Triangle** or **Diamond** from the dropdown.

The following diagram represents the radial gauge with the **Marker** pointer type, **Diamond** marker style, and **Cross** pointer placement.

![Pointer Type](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/pointer-property.png)

Marker length

This property is used to assign the **pointer marker length** value. You can also set the marker length property value based on dynamic values using [expressions](#).

Enable cap

To show or hide the pointer cap in radial gauge, toggle the **Enable Cap** checkbox, the pointer cap is supported only for pointer of type **Needle**.

You can customize the Cap in the **Advanced Options** menu. To open and handle the nested properties of Pointer Cap, refer to the [Advanced Properties](#) section.

![Pointer Cap](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/pointer-cap.png)

Using these properties, the pointer cap width, pointer cap color can be customized in the radial gauge design.

Width

This property is used to assign the **pointer width** value. You can also set the width property value based on dynamic values using [expressions](#).

Color

Using the color property, you can customize the **pointer color** of a radial gauge.

Scale

Angle

This property is used to set the values of start angle, sweep angle of the radial gauge. You can also set the start angle and sweep angle property value based on dynamic values using [expressions](#).

Width

This property is used to assign the scale width value. You can also set the width property value based on dynamic values using [expressions](#).

Color

Using the color property, you can customize the scale color of the radial gauge.

Reverse direction

By clicking the Reverse Direction checkbox, you can reverse the scale direction from maximum to minimum.

![Scale](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/scale.png)

Labels

To show or hide the scale labels in radial gauge, toggle the Label checkbox.

You can customize the scale label in the Advanced Options menu. To open and handle the nested properties of labels, refer to the [Advanced Properties](#) section.

![Scale Labels](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/labels-advanced-options.png)

Using these properties, the scale label font color, font family, font size, and font style can be customized in the radial gauge design.

Show labels at the end

To show or hide the scale labels at the end of the scale, toggle the Show Labels At End checkbox.

Rotate labels

To Rotate the scale labels, toggle the Rotate Lables checkbox.

Placement

Scale label placement can be Inside, Outside, or Cross. You can select an from the dropdown.

Position

This property is used to set the PivotX and PivotY of radial gauge. You can also set the pivot property value based on dynamic values using [expressions](#).

Tick mark

This property is used to customize the major and minor ticks of the radial gauge.

Major tick and minor tick

To show or hide major and minor ticks in a scale, toggle the Major Tick and Minor Tick checkbox.

You can customize major and minor ticks in the Advanced Options menu. To open and handle the nested properties of major and minor ticks, refer to the [Advanced Properties](#) section.

![Major Tick](/static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/major-tick.png)

Using this property, you can customize major and minor ticks interval, color, length and width.

Major tick placement and minor tick placement

Major tick and minor tick placement can be **Inside**, **Outside**, or **Cross**. The placement can be selected from the dropdown.

Range

Using this property, the range placement, start value, end value, start width, end width, and color can be customized in the radial gauge design.

Enable range

To show or hide the range in radial gauge, toggle the **Enable Range** checkbox.

![Enable Range]/(static/assets/on-premise/images/report-designer/report-items/gauge/radial-gauge/enable-range.png)

Placement

Range placement can be **Inside**, **Outside**, or **Cross**. You can select any option from the dropdown.

Range

This property is used to assign the range **start** and **end** values of the radial gauge. You can also set the start and end properties value based on dynamic values using [expressions](#).

Width

This property is used to assign the range **start width** and **end width** values of the radial gauge. You can also set the start and end width properties value based on dynamic values using [expressions](#).

Color

Using the color property, you can customize the **range color** of a radial gauge.

Position

The position property is used to set the width, height, left and top position of the radial gauge in the report design. To handle these properties using properties panel, refer to the [Position](#) section.

Visibility

The visibility property is used to conditionally show or hide the radial gauge report item on report preview or export action. To set visibility of radial gauge item using properties panel, refer to the [Visibility](#) section.

Miscellaneous

Custom attributes

This property can be used to set the values for radial gauge custom properties. To assign values for custom properties using properties panel, refer to the [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for gauge report item using properties panel refer [Tooltip](#) section.

Data bar

Data bars are small, simple charts which can be used for showing trends and changes in data over time, especially over many periods.

They convey a lot of information in a little space, often used in table and matrices.

Add data bar to the report

1. The data bar report item is listed in the item panel under the **KPI** category.

![Data bar item in item panel](/static/assets/on-premise/images/report-designer/report-items/data-bar/item-panel.png)

2. Drag and drop the data bar report item into the design area or inside tablix cell from the item panel.

![Drag and drop data bar report item into design area](/static/assets/on-premise/images/report-designer/report-items/data-bar/item-drag.png)

3. After dropping the data bar item in the design area or inside tablix cell, the respective item properties will be listed in the properties panel.

![Data bar item with properties view](/static/assets/on-premise/images/report-designer/report-items/data-bar/designer-area.png)

Create data

To present data in the data bar, create a dataset and bind data to it. In this designing section, the following dataset query is used for dataset creation.

'sql

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
FROM Production.ProductSubcategory PS INNER JOIN  
Sales.SalesOrderHeader SOH INNER JOIN  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryId =  
P.ProductSubcategoryId INNER JOIN  
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryId  
'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Data bar needs a minimum of 1 value element to showcase data. The measure or expression field that you would like to analyze can be dropped into **Y Values** section. The dimension that you would like to categorize the measure, can be dropped onto **Columns** section. If you would like to categorize based on a series, then the respective dimension can be dropped onto **Rows** section in addition.

To configure data into data bar follow the below steps:

1. To bind data to a data bar report item, focus on that report item.
2. Click **Properties** icon in the configuration panel, the property pane opens. Now, switch to **DATA** tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/data-bar/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/data-bar/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the **Measures** section; other type of columns and dimension expressions are listed under the **Dimensions** section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/data-bar/measures-dimensions-category.png)

5. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/data-bar/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/data-bar/y-value-chart-design-view.png)

6. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/data-bar/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/data-bar/aggregation-settings-menu.png)

7. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/data-bar/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/data-bar/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into Row(s) section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/data-bar/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/data-bar/row-field-design-preview.png)

Properties

Refer to the [Properties panel](#) section before proceeding with the following properties.

General Settings

![Name property](/static/assets/on-premise/images/report-designer/report-items/data-bar/name-property.png)

Name

Name property can be used to provide an unique name to the data bar item in the report.

Basic Settings

![Chart Basic settings](/static/assets/on-premise/images/report-designer/report-items/data-bar/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported data bar types will be listed in the **Chart Type** property dropdown, you can switch the required data bar type based on the data.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/data-bar/chart-type.png)

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

![Choose series](/static/assets/on-premise/images/report-designer/report-items/data-bar/single-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property.

The marker properties will be applied to the selected series in the [Choose Series](#) drop-down.

Show Data Label

Data label can be added to a chart series by using the [Show Data Label](#) property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property.

![Chart data label](/static/assets/on-premise/images/report-designer/report-items/data-bar/data-label-design.png)

The data label properties will be applied to the selected series in the [Choose Series](#) drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset smart label properties, refer [Enable Smart Label](#) property.

![Chart smart label](/static/assets/on-premise/images/report-designer/report-items/data-bar/enable-smart-label.png)

To apply smart label properties, enable [Data Label](#) for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Format chart series border](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart series border design](/static/assets/on-premise/images/report-designer/report-items/data-bar/border-design.png)

The series border properties will be applied to the selected series in the [Choose Series](#) drop-down.

You can also set properties based on dynamic values, by using the [Expressions](#). Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Dynamic chart series border](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose series in the [Choose Series](#) drop-down and choose color in [Series Color](#) property palette. Now, the selected color will be applied to the respective series in the chart design.

![Chart series color](/static/assets/on-premise/images/report-designer/report-items/data-bar/series-color.png)

You can also apply series color based on dynamic values, by using the [Expressions](#). Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/data-bar/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Chart appearance](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design. These properties are listed under **Chart Area** category.

![Chart Area](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for data bar report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The **PageName** property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Position

Position property is used to set the width, height, left and top position of the data bar in the report design. To handle these properties using properties panel refer [Position](#) section.

![Chart postion](/static/assets/on-premise/images/report-designer/report-items/data-bar/position.png)

Visibility

The visibility property is used to conditionally show or hide the data bar report item on report preview or export action. To set visibility of data bar item using properties panel, refer to the [Visibility](#) section.

![Chart visibility](/static/assets/on-premise/images/report-designer/report-items/data-bar/visibility.png)

| | |
|-----------------------------------------|----------------|
| Design ssrs data bar report using table | Set expression |
|-----------------------------------------|----------------|

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for data bar report item using properties panel refer [Tooltip](#) section.

[Set expression](#)

An expression can be set to few properties of the data bar report item to process the property values based on expressions. To set expressions to the data bar report item properties, refer to the [Set Expression](#) section.

[Reset expression](#)

To [Reset](#) the expression applied to a property, refer to the [Reset Expression](#) section.

[Advanced properties](#)

A few properties of the data bar report item contains nested properties. To open and handle the nested properties, refer to the [Advanced Properties](#) section.

[Design ssrs data bar report using table](#)

This section describes the steps to design [Data bar](#) report using SSRS table report item.

[Create dataset](#)

The following dataset query is used for this [Data bar](#) report.

```
'sql
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +
DATENAME(qq, SOH.OrderDate) AS OrderQtr,SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales
FROM Production.ProductSubcategory PS INNER JOIN
Sales.SalesOrderHeader SOH INNER JOIN
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =
P.ProductSubcategoryID INNER JOIN
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),
PS.ProductSubcategoryID
'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Add data bar report item

1. Drag and drop table report item to the design area.

![Data bar item in item panel](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/add-table.png)

2. Assign [dataset](#) to the table.

![Assign dataset to table](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/assign-data.png)

3. Design a simple table design with two columns like below.

![Table report design](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/basic-table-design.png)

4. Click on **Details group** in **Row Groups** pane, now the respective tablix member properties will be listed in the properties panel.

![Group properties](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/select-detail-group.png)

5. Click on the **Set Groups...** button in the properties panel. Now, the **Grouping** dialog will be opened like below.

![Group dialog](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/group-dialog.png)

6. Click on **Add** button and select **SubCat** and click **OK**.

![Group data in table](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/group-data.png)

7. Now select the **Sales** text box and [assign the following expression](#) =Sum(Fields!Sales.Value) in the cell.

![Sum sales value](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/sum-sales-value.png)

8. Format the numbers produced by the **Sales** field, using **Format** property. Set the '\$#,0;('\$#,0) as value format property field.

![Format textbox value](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/format-sales-value.png)

9. Add a column to the right and name it as Sales Indicator.

![Add column in table](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/add-column.png)

10. Drag and drop the data bar report item into last cell of the table.

![Add data bar report item](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/add-databar.png)

Now, the report design will look like below.

![Basic design of data bar report](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/intial-design.png)

Assign data

1. Select the cell containing data bar report item and switch to DATA tab in properties panel.

![Assign data to chart](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/switch-data-assign.png)

2. Drag and drop Sales field into Y-Values section as shown below.

![Add series to data bar](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/assign-series-value.png)

Configure properties

1. Now, switch to the PROPERTIES tab in the properties panel.

2. Choose the Sales series in the Choose Series dropdown.

![Choose series in chart](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/choose-series.png)

3. Enable Data Label property checkbox.

![Chart data label](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/enable-data-label.png)

4. Format the numbers produced by the Data labels, using Format property. Under the Advanced Options, set the '\$#,0;('\$#,0) as value format property field.

![Format datalabel value](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/format-values.png)

5. In the below design background color and font styles are changed in table cells to improvise the report design.

![Format table design](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/format-report.png)

Refer the [Cell Properties](#) to style the table cell.

Report preview

On report preview, the difference in sales between each product category will be displayed like below,

![Data bar RDL report preview](/static/assets/on-premise/images/report-designer/report-items/data-bar/design/report-preview.png)

Sparkline

Sparklines are small, simple charts which can be used for showing trends and changes in data over time, especially over many periods.

They represents multiple data points, often used in table and matrices.

Add sparkline to the report

1. The sparkline report item is listed in the item panel under the **KPI** category.

![Sparkline item in item panel](/static/assets/on-premise/images/report-designer/report-items/sparkline/item-panel.png)

2. Drag and drop the sparkline report item into the design area or inside tablix cell from the item panel.

![Drag and drop sparkline report item into design area](/static/assets/on-premise/images/report-designer/report-items/sparkline/item-drag.png)

3. After dropping the sparkline item in the design area or inside tablix cell, the respective item properties will be listed in the properties panel.

![Sparkline item with properties view](/static/assets/on-premise/images/report-designer/report-items/sparkline/designer-area.png)

Create data

To present data in the sparkline, create a dataset and bind data to it. In this designing section, the following dataset query is used for dataset creation.

```
'sql
```

```
SELECT PC.Name AS ProdCat, PS.Name AS SubCat, DATEPART(yy, SOH.OrderDate) AS OrderYear, 'Q' +  
DATENAME(qq, SOH.OrderDate) AS OrderQtr, SUM(SOD.UnitPrice * SOD.OrderQty) AS Sales  
  
FROM Production.ProductSubcategory PS INNER JOIN  
Sales.SalesOrderHeader SOH INNER JOIN  
Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID INNER JOIN  
Production.Product P ON SOD.ProductID = P.ProductID ON PS.ProductSubcategoryID =  
P.ProductSubcategoryID INNER JOIN
```

```
Production.ProductCategory PC ON PS.ProductCategoryID = PC.ProductCategoryID  
GROUP BY DATEPART(yy, SOH.OrderDate), PC.Name, PS.Name, 'Q' + DATENAME(qq, SOH.OrderDate),  
PS.ProductSubcategoryID  
'
```

Refer [Create Data](#) section and create dataset using the above query. AdventuresWorks database is used here.

Assign data

Sparkline needs a minimum of 1 value element to showcase data. The measure or expression field that you would like to analyze can be dropped into Y Values section. The dimension that you would like to categorize the measure, can be dropped onto Columns section. If you would like to categorize based on a series, then the respective dimension can be dropped onto Rows section in addition.

To configure data into sparkline follow the below steps:

1. To bind data to a sparkline report item, focus on that report item.
2. Click Properties icon in the configuration panel, the property pane opens. Now, switch to DATA tab.

![Chart properties pane](/static/assets/on-premise/images/report-designer/report-items/sparkline/chart-properties-pane.png)

3. The available data in the report will be listed in the drop-down, choose a data in the drop-down list.

![Choose the dataset for chart](/static/assets/on-premise/images/report-designer/report-items/sparkline/data-assign-drop-down.png)

4. The numeric columns and numeric expressions are listed under the Measures section; other type of columns and dimension expressions are listed under the Dimensions section.

![Measures and dimensions](/static/assets/on-premise/images/report-designer/report-items/sparkline/measures-dimensions-category.png)

5. Drag and Drop Measure Element:

Select and drag the numeric column (measure element) or the numeric expression column from the Measure section and drop it in the Y Values section.

![Add a Y-value field](/static/assets/on-premise/images/report-designer/report-items/sparkline/add-y-values-field.png)

Now, the report item design will look like below:

![Preview after adding y-value field](/static/assets/on-premise/images/report-designer/report-items/sparkline/y-value-chart-design-view.png)

6. Aggregate Options:

Click the **Settings** icon (highlighted below) to open the aggregation type drop-down list.

![Aggregate settings icon](/static/assets/on-premise/images/report-designer/report-items/sparkline/aggregation-settings-icon.png)

You can set the aggregation type by which you can compute the selected column.

![Aggregate menu list](/static/assets/on-premise/images/report-designer/report-items/sparkline/aggregation-settings-menu.png)

7. Drag and Drop Dimension Element:

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Add dimension field](/static/assets/on-premise/images/report-designer/report-items/sparkline/add-field-to-column-section.png)

Now, the report item design will look like below:

![Preview after adding dimension field](/static/assets/on-premise/images/report-designer/report-items/sparkline/column-field-design-preview.png)

8. Grouping:

You can group the added column element with another column, by adding the respective dimension element into **Row(s)** section.

![Achieve grouping by row values](/static/assets/on-premise/images/report-designer/report-items/sparkline/add-field-to-rows-section.png)

Now, the report item design will look like below.

![Preview of row value grouping](/static/assets/on-premise/images/report-designer/report-items/sparkline/row-field-design-preview.png)

Properties

Refer to the [Properties panel](#) section before proceeding with the following properties.

General Settings

![Name property](/static/assets/on-premise/images/report-designer/report-items/sparkline/name-property.png)

Name

Name property can be used to provide an unique name to the sparkline item in the report.

Basic Settings

![Chart Basic settings](/static/assets/on-premise/images/report-designer/report-items/sparkline/basic-settings.png)

Note: The properties under basic setting will be enabled in the chart properties panel, after assigning the data to the chart.

Chart Type

Supported line chart types will be listed in the **Chart Type** property dropdown, you can switch the required sparkline type based on the data.

![Chart Types](/static/assets/on-premise/images/report-designer/report-items/sparkline/chart-type.png)

Choose Series

You can add multiple series to the chart and the available series will be listed in the **Choose Series** drop-down. To customize the series appearance choose the required series name in the drop-down.

![Choose series](/static/assets/on-premise/images/report-designer/report-items/sparkline/single-series-list-in-drop-down.png)

Show Marker

Data markers are used to provide information about the data point to the user. You can add a shape and label to adorn each data point. To set/reset marker properties, refer [Show Marker](#) property.

The marker properties will be applied to the selected series in the **Choose Series** drop-down.

Show Data Label

Data label can be added to a chart series by using the **Show Data Label** property. The labels appear at the top of the data point, by default. To set/reset data label properties, refer [Show Data Label](#) property.

![Chart data label](/static/assets/on-premise/images/report-designer/report-items/sparkline/data-label-design.png)

The data label properties will be applied to the selected series in the **Choose Series** drop-down.

Enable Smart Label

Smart labels manage overlapping of labels even when a large number of labels are placed in close vicinity. To set/reset smart label properties, refer [Enable Smart Label](#) property.

![Chart smart label](/static/assets/on-premise/images/report-designer/report-items/sparkline/enable-smart-label.png)

To apply smart label properties, enable **Data Label** for chart data region.

Series Border

Series border properties can be used to customize the chart series border in the design.

![Format chart series border](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/series-border.png)

In the below design, border color, width and style properties are applied to the chart series.

![Chart series border design](/static/assets/on-premise/images/report-designer/report-items/sparkline/border-design.png)

The series border properties will be applied to the selected series in the **Choose Series** drop-down.

You can also set properties based on dynamic values, by using the **Expressions**. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

![Dynamic chart series border](/static/assets/on-premise/images/report-designer/report-items/chart/chart-series/border-expression-menu.png)

Series Color

Series Color property can be used to customize the series colors in the chart area. If the chart has multiple series, you can differentiate the series using this property.

Choose series in the **Choose Series** drop-down and choose color in **Series Color** property palette. Now, the selected color will be applied to the respective series in the chart design.

![Chart series color](/static/assets/on-premise/images/report-designer/report-items/sparkline/series-color.png)

You can also apply series color based on dynamic values, by using the **Expressions**. Refer [Set Expressions](#) and [Reset Expressions](#) section to open set/reset expression menu in properties panel.

Link To

You can configure **Hyperlink** or a **Report path** in the chart series to create an interactive report. Refer [Linking](#) section to set or reset link property for chart series.

![Link To property](/static/assets/on-premise/images/report-designer/report-items/sparkline/link-to-property.png)

Appearance

The border style, color, width and background color properties can be used to style the chart and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

![Chart appearance](/static/assets/on-premise/images/report-designer/report-items/chart/appearance-property.png)

Chart Area

Chart Area properties such as border width, color, and background color can be used to customize the area of the chart design. These properties are listed under **Chart Area** category.

![Chart Area](/static/assets/on-premise/images/report-designer/report-items/chart/chart-area.png)

Page break

The page break property can be used to control the amount of information on each page when you preview the report. Follow the below steps to apply page break property for sparkline report item.

1. The Break Location property specifies where the page break should occur. Choose any **Break Location** type in the drop-down.

![Break location](/static/assets/on-premise/images/report-designer/report-items/rectangle/break-location-types.png)

2. To restart the page numbering on each page, enable **Page Number Reset** property checkbox.

![Reset page number](/static/assets/on-premise/images/report-designer/report-items/rectangle/page-break-property.png)

3. The PageName property specifies a page id for the new page that the page break causes. You can specify a static text or dynamic expression to this property.

Position

Position property is used to set the width, height, left and top position of the sparkline in the report design. To handle these properties using properties panel refer [Position](#) section.

![Chart position](/static/assets/on-premise/images/report-designer/report-items/sparkline/position.png)

Visibility

The visibility property is used to conditionally show or hide the sparkline report item on report preview or export action. To set visibility of sparkline item using properties panel, refer to the [Visibility](#) section.

![Chart visibility](/static/assets/on-premise/images/report-designer/report-items/sparkline/visibility.png)

Miscellaneous

Custom Attributes

This property can be used to set the values for chart custom properties. To create and assign values for custom properties using properties panel refer [Custom Properties](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for sparkline report item using properties panel refer [Tooltip](#) section.

Set expression

An expression can be set to few properties of the sparkline report item to process the property values based on expressions. To set expressions to the sparkline report item properties, refer to the [Set Expression](#) section.

Reset expression

To [Reset](#) the expression applied to a property, refer to the [Reset Expression](#) section.

Advanced properties

A few properties of the sparkline report item contains nested properties. To open and handle the nested properties, refer to the [Advanced Properties](#) section.

Design ssrs sparkline report using table

This section describes the steps to design Sparkline report using SSRS table report item.

Create dataset

The following dataset query is used for this Sparkline report.

```
`sql
```

```
SELECT Prodcat.EnglishProductName,
prodSubcat.EnglishProductSubcategoryName,
prod.EnglishProductName,
```

```
prod.Color,  
fact.[TotalProductCost],  
fact.SalesAmount,  
fact.TaxAmt,  
fact.[OrderDate],  
fact.[ShipDate]  
FROM dbo.DimProduct as prod  
INNER JOIN dbo.DimProductSubcategory AS prodSubcat ON  
prod.ProductSubcategoryKey = prodSubcat.ProductSubcategoryKey  
INNER JOIN dbo.DimProductCategory AS Prodcat ON  
prodSubcat.ProductCategoryKey = Prodcat.ProductCategoryKey  
INNER JOIN dbo.FactInternetSales AS fact ON  
fact.ProductKey = prod.ProductKey  
\
```

Refer [Create Data](#) section and create dataset using the above query. AdventureWorksDW2014 database is used here.

Add sparkline report item

1. Drag and drop table report item to the design area.

![Sparkline item in item panel](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/add-table.png)

2. Assign [dataset](#) to the table.

![Assign dataset to table](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/assign-data.png)

3. Assign [EnglishProductSubcategoryName](#) field in the first cell as shown below.

![Assign value in table cell](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/add-group-value.png)

4. Click on [Details group](#) in [Row Groups](#) pane, now the respective tablix member properties will be listed in the properties panel.

![Group properties](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/select-detail-group.png)

5. Click on the **Set Groups...** button in the properties panel. Now, the **Grouping** dialog will be opened like below.

![Group dialog](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/group-dialog.png)

6. Click on **Add** button and select **EnglishProductSubcategoryName** and click **OK**.

![Group data in table](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/group-data.png)

7. Assign **Sales** field in the second cell as shown below.

![Assign value in table cell](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/assign-sales-field.png)

8. Now select the **Sales** text box and [assign the following expression](#)
`=Sum(Fields!SalesAmount.Value)` in the cell.

![Sum sales value](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/sum-sales-value.png)

9. Drag and drop the sparkline report item into last cell of the table as shown below.

![Add Sparkline item in table](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/add-sparkline.png)

Now, the report design will look like below.

![Basic design of sparkline report](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/intial-design.png)

Assign data

1. Select the cell containing sparkline report item and switch to **DATA** tab in properties panel.

![Assign data to chart](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/switch-data-assign.png)

2. Drag and drop **TotalProductCost** field into **Y-Values** section as shown below.

![Add series to sparkline](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/assign-series-value.png)

3. Drag and drop **OrderDate** field into **Column** section as shown below.

![Add column to sparkline](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/assign-column-value.png)

Now, the report design will look like below.

![Basic design of sparkline report](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/assign-data-design.png)

Configure properties

1. Now, switch to the **PROPERTIES** tab in the properties panel.
2. Choose the **Sales** series in the **Choose Series** dropdown.

![Choose series in chart](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/choose-series.png)

3. Enable **Data Label** property checkbox.

![Chart data label](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/enable-data-label.png)

4. Format the numbers produced by the **Data labels**, using **Format** property. Under the **Advanced Options**, set the '\$#,0;('\$#,0) as value format property field.

![Format datalabel value](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/format-values.png)

5. In the below design background color and font styles are changed in table cells to improvise the report design.

![Format table design](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/format-report.png)

Refer the [Cell Properties](#) to style the table cell.

Report preview

On report preview, the report is showing sales amount progress from start year to till end.

![Sparkline RDL report preview](/static/assets/on-premise/images/report-designer/report-items/sparkline/design/report-preview.png)

Barcode

A barcode report item is used for representing data in a visual, machine-readable form. You can build reports that display a barcode to make tasks like tracking shipping orders and employee identification numbers easier.

One-Dimensional barcodes

One-Dimensional barcodes represents data in the widths and spacings of printed parallel lines, which is why they are also called as linear barcodes or symbolics. Linear barcodes are read in one direction (horizontally). Linear symbolics allows the coding of small amounts of information content (a maximum of 20-30 digits or symbols).

Adding a one dimensional barcode to the report

1. One-dimensional barcode report item is listed in the item panel under the **Barcodes** category.

![Barcodes listed in item panel](/static/assets/on-premise/images/report-designer/report-items/barcode/barcode-item-in-item-panel.png)

2. Drag and drop the 1-D barcode report item into the design area from the item panel.

![Drag and drop barcode report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/drag-and-drop-1D-barcode.png)

3. Once you drop the barcode item into design area, respective item properties will be listed the properties panel.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-barcode-properties.png)

Properties

Properties required to generate 1D barcodes are listed under the **Basic Settings** category in the properties panel.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-basic-settings.png)

Symbology Type

The list of supported 1D barcode symbologies are provided in the **Symbology Type** drop-down list.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/symbology-types.png)

To change the type of existing 1D barcode in the report, select the specific barcode item and choose the symbology type in the drop-down. Now, the respective 1D barcode type will be rendered in the report design.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/symbology-design.png)

Text

Text property can be used to specify the value to be encoded in the barcode.

Each barcode type requires different type and length of input values for encoding, so provide the valid values based on the respective 1D barcode type. Supported barcode types and their valid input values are listed in the [Supported Barcode Types](#) section.

If you provide invalid values, the error message will be displayed in the barcode item as below,

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/invalid-encoding.png)

You can also encode the values in barcodes dynamically by using the **Expressions**.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-barcode-expression-menu.png)

In the expression builder, create expression using built-in-functions or data fields and click **OK**.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-expression-demo.png)

Now, the expression value will be encoded in the barcode at run-time. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Text Visibility

The **Text Visibility** property can be used to show/hide the label text of the barcode. The label shows a value of the barcode. In the below snaps, the **Code39** barcode with and without label text is displayed.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-label-visible.png)

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-label-hidden.png)

Two-Dimensional barcodes

Two-dimensional (2D) barcodes are used for coding large amounts of information in a bar code, potentially up to several pages worth. Such a barcode would consist of square cells, dots, hexagons, and other geometrical figures.

Adding a two dimensional barcode to the report

1. Two-dimensional barcode report item is listed in the item panel under the **Barcodes** category.

![Barcodes listed in item panel](/static/assets/on-premise/images/report-designer/report-items/barcode/2D-barcode-item-in-item-panel.png)

2. Drag and drop the 2-D barcode report item into the design area from the item panel.

![Drag and drop barcode report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/drag-and-drop-2D-barcode.png)

3. Once you drop the barcode item into design area, respective item properties will be listed in the properties panel.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/2D-barcode-properties.png)

Properties

Properties required to generate QR barcode item are listed under the **Basic Settings** category in the properties panel.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/2D-basic-settings.png)

Text

Text property can be used to specify the value to be encoded in the QR barcode.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/2D-text-property.png)

You can also encode the values in barcodes dynamically by using the **Expressions**.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/expression-menu.png)

In the expression builder, create expression using built-in-functions or data fields and click **OK**.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/1D-expression-demo.png)

Now, the expression value will be encoded in the barcode at run-time. Refer [Set Expression](#) and [Reset Expression](#) section to open set/reset expression menu in properties panel.

Text Visibility

The **Text Visibility** property can be used to show/hide the label text of the barcode. The label shows a value of the barcode. In the below snaps, the QR barcode with and without label text is displayed.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/2D-label-visible.png)

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/2D-label-hidden.png)

Supported barcode types and their valid input values are listed in the [Supported Barcode Types](#) section.

General properties

Name

Name property can be used to provide an unique name to the barcode item in the report.

Appearance

The border style, color, width and background color properties are used to style the barcode and customize its appearance in the report design. These properties are listed under the **Appearance** category in the properties panel.

Border

Border properties are used to add or customize the border around a barcode item to visually separate it in the report design. To set border properties to the tablix item using properties panel refer [Border Properties](#) section.

Background color

Using the background color property you can color the barcode background. To set background color using properties panel refer [Background color](#) section.

![Barcode item with properties view](/static/assets/on-premise/images/report-designer/report-items/barcode/appearance.png)

Position

Position property is used to set the width, height, left and top position of the barcode in the report design. To handle these properties using properties panel refer [Position](#) section.

| | |
|--------------------------------------|----------------|
| Design ssrs rdl report using barcode | Set expression |
|--------------------------------------|----------------|

Visibility

Visibility property is used to conditionally show or hide the barcode report item on report preview or export action. To set visibility of barcode item using properties panel refer [Visibility](#) section.

Tooltip

Tooltip property can be used to display informative text or value, when the user hovers over on the report item in report preview. To set tooltip for barcode item using properties panel refer [Tooltip](#) section.

Set expression

An expression can be set to few properties of the tablix report item to process the property values based on expressions. To set expressions to the tablix report item properties, refer [Set Expression](#) section.

Reset expression

To **Reset** the expression applied to a property, refer [Reset Expression](#) section.

Advanced properties

Few properties of the tablix report items contains nested properties. To open and handle nested properties, refer [Advanced Properties](#) section.

Design ssrs rdl report using barcode

This section describes the steps to design **Product Details** report using barcode and SSRS table report item.

Create dataset

To present data in the table format, create a dataset and bind data to the table data region. In this designing section, the following dataset query is used for dataset creation.

``sql`

```
SELECT TOP 100 [Order Details].ProductID, Products.ProductName, [Order Details].Quantity, [Order Details].UnitPrice, [Order Details].Discount
FROM [Order Details] INNER JOIN
Products ON [Order Details].ProductID = Products.ProductID
```

`,`

Refer [Create Data](#) section and create dataset using the above query. **Northwind** database is used here.

Add a table to the report

1. Drag and drop the table report item into the design area from the item panel.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/add-table-report-item.png)

2. Assign data to the table using **Dataset** property in the table properties.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/assign-data-to-table.png)

Configure table

1. Add the required table column headers in the table as shown in the below snap.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/table-cell-headers.png)

2. Now, assign the required product details data fields in the **Details** group row of the table cell.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/assign-fields-in-table.png)

Following expressions are used in the **UnitPrice** and **Total Price** cells of the table.

- `=FormatCurrency(FormatNumber(Fields!UnitPrice.Value,2))`
- `=FormatCurrency(FormatNumber(Fields!Quantity.Value * Fields!UnitPrice.Value,2))`

Add barcode report item

In the above table design, **Product ID** cell is left empty, it would be better if we present the product ID using the barcode.

1. Drag a rectangle item and drop into the **Product ID** column cell.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/add-rectangle-in-table.png)

2. Now, drag and drop it into the **Product ID** column cell.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/add-barcode-in-table.png)

3. The barcode report item will be placed inside the rectangle as shown below,

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/barcode-item-with-rectangle.png)

If we directly add barcode into table cell, the report item fill the whole cell. So, rectangle report item is used as container.

Encode value in barcode

1. Select the barcode report item and open properties panel,

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/barcode-properties-with-table.png)

2. Open the expression menu in the **Text** property under **Basic Settings** category.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/open-expression-builder.png)

3. In the **Expression Builder** assign the **=Fields!ProductID.Value** field value and click **Ok**.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/assign-expression.png)

Now, the **Product ID** of each product will be encoded in the barcode report item on report export action.

Customize appearance

Using the position and appearance properties, you can customize the barcode report item as shown below,

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/resize-barcode-with-table.png)

Format table

1. Select the first row, set the background and font properties in the properties panel.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/customize-text-design.png)

2. Resize the table to the required size for better report design.

![Drag and drop tablix report item into design area](/static/assets/on-premise/images/report-designer/report-items/barcode/table-final-design.png)

Report header

1. Enable the report **Header** to add a title to the report.

![Enable header tag](/static/assets/on-premise/images/report-designer/report-items/barcode/enable-report-header.png)

Refer [Show or hide header and footer](#) section to add or remove header/footer in the report.

2. Now, add a rectangle report item in the report header area and a textbox within the rectangle.

![Add report items in header area](/static/assets/on-premise/images/report-designer/report-items/barcode/add-report-items-in-header-area.png)

3. Set the report title text in the textbox and customize the appearance of the title using the textbox and rectangle properties in properties panel as required.

![Report title text](/static/assets/on-premise/images/report-designer/report-items/barcode/report-title-text.png)

Final design

A final design of Product Details report will look like below.

![Product Details report design](/static/assets/on-premise/images/report-designer/report-items/barcode/product-details-final-design.png)

Report preview

On report preview, the product details report will be displayed like below,

![Product Details report preview](/static/assets/on-premise/images/report-designer/report-items/barcode/report-preview.png)

Supported barcode types

The following table contains the supported types and associated valid characters.

| Symbol | Supported characters | Length |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| <u>QR Code</u> | [0-9]; [A-Z (upper-case only)]; [space \$ % * + - . / , :]; [Shift JIS characters] | variable |
| <u>Code 39</u> | [0-9]; [A-Z]; [- . \$ / + % SPACE] | variable |
| <u>Code 39 Extended</u> | [0-9]; [A-Z]; [a-z] | variable |
| <u>Code 11</u> | [0-9]; [-] | variable |
| <u>Codabar</u> | [0-9]; [- \$: / . +] | variable |
| <u>Code 93</u> | [0-9]; [A-Z]; [- . \$ / + % SPACE] | variable |
| <u>Code 128A</u> | [0-9]; [A-Z]; [NUL (0x00) SOH (0x01) STX (0x02) ETX (0x03) EOT(0x04) ENQ (0x05) ACK (0x06) BEL (0x07) BS (0x08) HT (0x09) LF (0x0A) VT(0x0B) FF (0x0C) CR (0x0D) SO (0x0E) SI (0x0F) DLE (0x10) DC1 (0x11) DC2(0x12) DC3 (0x13) DC4 (0x14) NAK (0x15) SYN (0x16) ETB (0x17) CAN(0x18) EM (0x19) SUB (0x1A) ESC (0x1B) FS (0x1C) GS (0x1D) RS (0x1E) US(0x1F) SPACE (0x20)]; [" ! # \$ % & ' () * + , - . / ; < = > ? @ [/] ^ _] | variable |
| <u>Code 128B</u> | [0-9]; [A-Z]; [a-z]; [SPACE (0x20) ! " # \$ % & ' () * + , - . / ; < = > ? @ [/] ^ _ { } ~ DEL (•)] | variable |
| <u>Code 128C</u> | ASCII 00-99(encodes each two digit with one code) | variable |
| <u>UPCA</u> | numeric [0..9] | 12 digits |
| <u>EAN-13</u> | numeric [0..9] | 12 digits |
| <u>EAN-8</u> | numeric [0..9] | 7 usable digits |
| <u>Code39 Mod 43</u> | [A-Z (upper-case only)] [0-9] [SPACE - . \$ / + %] | variable |
| <u>Interleaved 2 of 5</u> | numeric [0..9] | variable |

Open report

From Device

| | | |
|-----------------|-----------------------------|----------|
| Standard 2 of 5 | numeric [0..9] | variable |
| Pharmacode | numeric [0..9] and generic; | variable |

Open report

To open an existing report, click on the **Open** icon in the designer toolbar.

![Open report menu](/static/assets/on-premise/images/report-designer/open-report/open-report-menu.png)

From Device

1. To open the report from local computer, click on **From Device** option in the context menu. Now, the client browse dialog will be launched.

![Open report from local computer](/static/assets/on-premise/images/report-designer/open-report/client-browse-dailog.png)

2. Select the report and click on **Open** button.

From Server

1. To open the report from server, click on **From Server** option in the context menu. Now, the server browse dialog will be launched.

![Open report from server](/static/assets/on-premise/images/report-designer/open-report/server-browse-dialog.png)

2. Browse to the report server reports folder location.

![Browse reports in report server](/static/assets/on-premise/images/report-designer/open-report/browse-server-reports.png)

3. Select the report and click on **Open** button.

Save a Report

In Web Report Designer, you can save a report to your computer or publish report into Report Server.

Open the save menu in toolbar and click on **Save** option in the context menu.

![Save report into report server](/static/assets/on-premise/images/report-designer/save-report/directly-save-report-to-the-server.png)

If you have opened a report from server, the **Save** option will directly replace the report into **server**.

If you have created a new report, the **Save** option will download the report to the **local computer**.

Save a report into report server

1. Open the save menu in toolbar.

![Save menu in web designer](/static/assets/on-premise/images/report-designer/save-report/save-menu.png)

2. In **Save As** option, click on **To Server**. It will launch the **Save As Report** dialog.

![Save a new report into report server](/static/assets/on-premise/images/report-designer/save-report/save-as-report-dialog.png)

3. Browse to the reports folder location and provide name for the report in **Name** field.

![Browse the reports folder in report server](/static/assets/on-premise/images/report-designer/save-report/browse-folder-and-save-report.png)

4. Click on **Save** button to save the report.

5. If you are saving the report with duplicate report name, it will launch **Save As Report** alert dialog,

![Replace report alert](/static/assets/on-premise/images/report-designer/save-report/replace-existing-report-alert.png)

6. Click **Yes** to replace the report or **No** to edit the report name in the name field.

Save report to your computer

1. After designing the report, open the save menu in toolbar. In **Save As** option, click on **To Device**.

![Save report to device](/static/assets/on-premise/images/report-designer/save-report/save-to-device-menu.png)

2. Now, the report will be automatically downloaded to the local computer.

Preview a Report

1. After successfully designing a report in Web report designer, to preview the report click on the **Preview** button in the top-right corner of the report header.

![Preview button in report designer](/static/assets/on-premise/images/report-designer/preview-report/preview-button-location.png)

2. Now, the report will be previewed using the inbuilt Report Viewer.

![Report preview using Report Viewer](/static/assets/on-premise/images/report-designer/preview-report/report-preview-in-report-viewer.png)

Supported and Unsupported Items in Web Report Designer

Report Items

The following table shows the supported report items in the Web Report Designer.

| Features | Supported |
|----------------|-----------|
| Textbox | Yes |
| Image | Yes |
| Rectangle | Yes |
| Line | Yes |
| Column Chart | Yes |
| Bar Chart | Yes |
| Pie Chart | Yes |
| Doughnut Chart | Yes |
| Pyramid Chart | Yes |
| Funnel Chart | Yes |
| Area Chart | Yes |
| Line Chart | Yes |
| Bubble Chart | Yes |
| Radar Chart | Yes |
| Polar Chart | Yes |
| SubReport | Yes |
| Table | Yes |
| Gauge | No |
| Sparkline | Yes |
| Maps | No |
| Data Bar | Yes |
| Indicator | Yes |
| Matrix | Yes |
| List | Yes |
| 3-D Charts | No |
| Sunburst | No |

Keyboard Shortcuts

Features

| | | |
|---------------------|-----|--|
| Range Charts | No | |
| Tree Map Chart | No | |
| Custom Report Items | Yes | |

Features

| | |
|-------------------|-----------|
| Features | Supported |
| ----- ----- | |
| Datasource | Yes |
| Dataset | Yes |
| Parameters | Yes |
| Embed Image | Yes |
| Code Modules | Yes |
| References | Yes |
| Variables | No |
| Built-in Fields | Yes |
| Report Properties | Yes |

DataSources

The following table lists the supported data sources for establishing a connection to the data provider and retrieve data in the Web Report Designer.

| | |
|-----------------------|-----------|
| DataSources | Supported |
| ----- ----- | |
| Microsoft SQL Server | Yes |
| ODBC | Yes |
| OLE DB | Yes |
| Oracle | Yes |
| SQL CE | Yes |
| XML | Yes |
| Extension Data Source | Yes |

Keyboard Shortcuts

Bold Report Designer allows you to design a report using the keyboard (without mouse). You can use the following shortcut keys to work with the web designer.

Shortcut keys

| Action | Shortcut Key |
|-------------|--------------|
| Open Report | Ctrl + O |

How to startup the Bold Reports On-Premise Edition using the newly added bindings from IIS Shortcut keys

| | |
|-----------------------------------------|----------------------------|
| Save Report | Ctrl + S |
| Redo | Ctrl + Y |
| Undo | Ctrl + Z |
| Copy | Ctrl+ C |
| Paste | Ctrl+ V |
| Zoom in | Ctrl+ - |
| Zoom out | Ctrl+ + |
| Select all report item | Ctrl+ A |
| Remove all report item selection | Esc |
| Delete selected report item | Del |
| To move selected report item | Arrow keys |
| To increase selected report item width | Ctrl + Shift + Right Arrow |
| To decrease selected report item width | Ctrl + Shift + Left Arrow |
| To increase selected report item height | Ctrl + Shift + Up Arrow |
| To decrease selected report item height | Ctrl + Shift + Down Arrow |
| To resize selected report item | Shift + Arrow keys |
| To Enable/Disable Header | Alt + H |
| To Enable/Disable Footer | Alt + F |
| To navigate report items | Tab |
| To navigate report items | Shift + Tab |

How to startup the Bold Reports On-Premise Edition using the newly added bindings from IIS

You can change the Bold Reports URL in Config.xml file of Report Server.

The following steps to change the Bold Reports URL:

1. Go to the deployed location and update the new binding values in following configuration files.

By default, Bold Reports will be deployed on C:\Bold Reports

- Update the **InternalAppReportUrl** value in config file from below location.

{Deployed Location}\IDP\UMS\Configuration\boldreports\Config.xml

![IDP Config File](/static/assets/on-premise/images/getting-started/idp-config.png)

- Update the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file from below location.

{Deployed Location}\Report Server\Configuration\Config.xml

![RS Config File](/static/assets/on-premise/images/getting-started/rs-config.png)

2. Restart the site in IIS and browse the site. Configure the Bold Reports On-Premise Edition startup by referring this [link](#).

How to map a domain name for Bold Report On-Premise Edition before startup

Before configuring the Bold Reports On-Premise Edition, you must change the Bold Reports URL in **Config.xml** file of Bold Reports On-Premise Edition.

The following steps to change the Bold Reports URL:

1. Go to the deployed location and update the new binding values in following configuration files.

By default, Bold Reports will be deployed on C:\Bold Reports

- Update the **InternalAppReportUrl** value in config file from below location.

{Deployed Location}\IDP\UMS\Configuration\boldreports\Config.xml

![IDP Config File](/static/assets/on-premise/images/getting-started/idp-config.png)

- Update the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file from below location.

{Deployed Location}\Report Server\Configuration\Config.xml

![RS Config File](/static/assets/on-premise/images/getting-started/rs-config.png)

2. Restart the site in IIS and browse the site. Configure the Bold Reports On-Premise Edition startup by referring this [link](#)

How to resolve unavailable error with Bold Reports On-Premises Edition when it is occurred in reason of domain name change

If Bold Reports On-Premise site fails to launch, you have to change the Bold Reports URL in **Config.xml** file of Bold Reports On-Premise Edition.

The following steps to change the Bold Reports URL:

1. Go to the deployed location and update the new binding values in following configuration files.

By default, Bold Reports will be deployed on C:\Bold Reports

- Update the **InternalAppReportUrl** value in config file from below location.

{Deployed Location}\IDP\App_Data\Configuration\Config.xml

![IDP Config File](/static/assets/on-premise/images/getting-started/idp-config.png)

- Update the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file from below location.

{Deployed Location}\Report Server\Configuration\Config.xml

- If file exists in below location, change the **InternalAppDataServiceUrl**, and **InternalAppIdpUrl** values in the config file

{Deployed Location}\Report Server\App_Data\Configuration\Config.xml

![RS Config File](/static/assets/on-premise/images/getting-started/rs-config.png)

2. Restart the site in IIS and browse the site.
3. Delete the site by clicking the "Delete" option from site listing page.

![Delete option in site](/static/assets/on-premise/images/how-to/delete-option-in-site.png)

4. Create a new site by following this [link](#)

How to setup the staging and production environment for Bold Reports On-Premise Edition

This section explains how to set up Bold Reports On-Premise Edition in staging and production environment.

Prepare a staging environment

- Install Bold Reports On-Premise build by following the [Installation link](#).
- After the installation completed, configure the Bold Reports On-Premise Edition by following the [application startup link](#).

Configure the Bold Reports with **Microsoft SQL Server** credentials and that should be accessible for other machines.

PostgreSQL database support is provided on Bold Reports latest version 2.2.23.

- Start the Bold Reports On-Premise Edition and test all the features in Bold Reports On-Premise.

Staging to production

Copy resources

Bold Reports application resources will be available in C:\Bold Reports location. We have to copy all the resources except Utilities folder to production server or the location will be used for production in same server.

Create a Bold Reports Site

IDP application should be added as main site for Bold Reports and it is used to authenticate the users in Bold Reports.

- Go to the IIS, right click the Sites and click Add Website.

![Add Website](/static/assets/on-premise/images/how-to/add-website.png)

- Fill the following details for adding IDP application.

Site Name - Provide the name of the site for production server.

Physical path - IDP folder is need to be provided.

Example: Extracted location on production machine is D:\Bold Reports, then refer the IDP folder path D:\Bold Reports\IDP.

Bindings - Provide the same bindings http or https, which was configured on staging machine.

Example: In staging machine, if you have bindings as http and port number 80, include same bindings on the production machine site.

![Fill Details in Add Website](/static/assets/on-premise/images/how-to/fill-detatils-in-add-website.png)

Add a reporting application

- Reporting application is a report server application and it should added as sub-application for IDP application.
- Right click the added site to add reporting application.

![reporting](/static/assets/on-premise/images/how-to/reporting.png)

- Fill the following details for adding reporting application.

Alias - Provide the Alias as reporting.

Physical path - Refer the Report Server folder path from the extracted location on production machine.

Example: Extracted location on production machine is D:\Bold Reports, then refer the Report Server folder path D:\Bold Reports\Report Server.

API application for Bold Reports IDP

- Right click the added site and click Add Application.
- Fill the following details as shown in figure and click ok, application will be added to the site.

![IDP-API](/static/assets/on-premise/images/how-to/idp-api.png)

Alias - Provide the Alias as **api**.

Physical path - Refer the IDP API folder path from the extracted location on production machine.

Example: Extracted location on production machine is **D:\Bold Reports**, then refer the IDP API folder path **D:\Bold Reports\IDP\API**.

UMS application for Bold Reports IDP

- UMS application is an user management application for Bold Reports and it should added as sub-application for IDP application.
- Right click the site and click **Add Application**.
- Fill the following details as shown in figure and click **ok**, application will be added to the site.

![IDP-UMS](/static/assets/on-premise/images/how-to/idp-ums.png)

Alias - Provide the Alias as **ums**.

Physical path - Refer the IDP UMS folder path from the extracted location on production machine.

Example: Extracted location on production machine is **D:\Bold Reports**, then refer the IDP UMS folder path **D:\Bold Reports\IDP\UMS**.

Windows Authentication application for Bold Reports IDP

- Windows Authentication application is used to authenticate windows directory users in Bold Reports and it should added as sub-application for IDP application.
- Right click the site and click **Add Application**.
- Fill the following details as shown in figure and click **ok**, application will be added to the site.

![IDP Windowsauthentication](/static/assets/on-premise/images/how-to/idp-windowsauthentication.png)

Alias - Provide the Alias as **ums**.

Physical path - Refer the IDP Windows Authentication folder path from the extracted location on production machine.

Example: Extracted location on production machine is **D:\Bold Reports**, then refer the IDP Windows Authentication folder path **D:\Bold Reports\IDP\WindowsAuthentication**.

API application for reporting application

- Right click the **reporting** application and click **Add Application**.
- Fill the following details as shown in figure and click **ok**, application will be added to the site.

![Report Server API](/static/assets/on-premise/images/how-to/reporting-api.png)

Alias - Provide the Alias as **api**.

Physical path - Refer the Report Server API folder path from the extracted location on production machine.

How to Set up Azure Active Directory to perform authentication using Single Sign-On for Bold Reports On-Premise

Steps to set up Azure Active Directory for Bold Reports On-Premise

Example: Extracted location on production machine is D:\Bold Reports, then refer the Report Server API folder path D:\Bold Reports\Report Server\API.

Jobs application for reporting application

Jobs application is necessary to sent schedule mails to the users and it should be added as sub-application for reporting application.

- Right click the reporting application and click Add Application.
- Fill the following details as shown in figure and click ok, application will be added to the site.

![Report Server Jobs](/static/assets/on-premise/images/how-to/reporting-jobs.png)

Alias - Provide the Alias as jobs.

Physical path - Refer the Report Server Jobs folder path from the extracted location on production machine.

Example: Extracted location on production machine is D:\Bold Reports, then refer the Report Server Jobs folder path D:\Bold Reports\Report Server\Jobs.

Report Service application for reporting application

Report Service application is necessary for rendering or designing the report in web designer and it should be added as sub-application for reporting application.

- Right click the reporting application and click Add Application.
- Fill the following details as shown in figure and click ok, application will be added to the site.

![Report Service](/static/assets/on-premise/images/how-to/reporting-reportservice.png)

Alias - Provide the Alias as report service.

Physical path - Refer the Report Server Report Service folder path from the extracted location on production machine.

Example: Extracted location on production machine is D:\Bold Reports, then refer the Report Server Report Service folder path D:\Bold Reports\Report Server\Report Service.

After completing all steps, browse the hosted Bold Reports site.

You can map a new domain for Bold Reports by referring the following [link](#)

How to Set up Azure Active Directory to perform authentication using Single Sign-On for Bold Reports On-Premise

This section explains on how to perform Single Sign-On for users in Azure Active Directory on Bold Reports On-Premise.

Steps to set up Azure Active Directory for Bold Reports On-Premise

Prerequisites

- An Azure account with Active Directory support.
- Install Bold Reports On-Premise and Login with Administrator account.

Setup Azure Active Directory application

Log on to the Azure portal to create an Azure Active Directory.

1. Click Create a resource and search Azure Active Directory as follows.

[\[Create a resource\] \(/static/assets/on-premise/images/how-to/create-resource-option.png\)](#)

![Create Directory 1](/static/assets/on-premise/images/how-to/create-resource.png)

2. Click Create in the following screenshot.

![Create Directory 1](/static/assets/on-premise/images/how-to/ad-create.png)

3. In the dialog box, enter the Name, Domain Name, and choose the Country or Region, and then click Create.

![Create Directory 2](/static/assets/on-premise/images/how-to/ad-create1-1.png)

- The application will be added to the directory and you can view the details of the application in the App registrations.

Go to the Azure Active Directory. In the directory, you should add two applications in which one acts as a Web API for authenticate the Bold Reports On-Premise, and an other application that acts as native client application for authenticate Bold Reports On-Premise mobile app.

Steps to register Bold Reports On-Premise application in Azure Active Directory

1. Enter into the created directory and click Azure Active Directory, and then select App registrations.
 2. Now, click New application registration to add a new application.

![Create Application 1](/static/assets/on-premise/images/how-to/add-application-1.png)

3. Enter the name of the application and choose the following options.

![Register](/static/assets/on-premise/images/how-to/register-option.png)

- Accounts in any organizational directory (Any Azure AD directory - Multitenant) as **Supported account types**.
 - Web under the **Redirect URI(optional)** section and enter the Redirect URI. And then click **Register**.

The application will be added to the directory and you can view the details of the application in the [App registrations](#).

4. Select Branding in the left side menu and enter the Home page URL and click Save.

How to Set up Azure Active Directory to perform authentication using Single Sign-On for Bold Reports On-Premise

Steps to set up Azure Active Directory for Bold Reports On-Premise

![Branding](/static/assets/on-premise/images/how-to/branding.png)

5. Select Authentication in the left side menu and save the Logout URL, Implicit grant and Supported account types as highlighted in the following screenshot.

![Authentication](/static/assets/on-premise/images/how-to/authentication.png)

- Select Certificates & secrets to add client secret by clicking the New client secret as in the following screenshot.

!{Certificates and secrets}(/static/assets/on-premise/images/how-to/certificates-and-secrets.png)

- Provide description and choose the expires option. Click on Add button.

![Client secret Durations](./static/assets/on-premise/images/how-to/client-secret-duration.png)

Save the client secret value generated.

7. Go to API permissions, click Add a permission and then click on Microsoft Graph.

[\[!Microsoft Graph\]\(/static/assets/on-premise/images/how-to/microsoft-graph.png\)](#)

| Application Permissions | |
|-------------------------|-------------------------------------------------------------|
| Directory | Read directory data |
| Delegated Permissions | |
| 1. Directory | Read directory data, Access directory as the signed in user |
| 2. Group | Read all groups |
| 3. User | Read and write access to user profile |
| 4. Profile | View user's basic profile |

Select the above listed permissions and click on **Update permissions** button.

![Update Permissions](/static/assets/on-premise/images/how-to/update-permissions.png)

8. Go to API permissions, click Add a permission and then click on Azure Active Directory Graph as shown below.

![Azure Active Directory Graph](/static/assets/on-premise/images/how-to/azure-active-directory-graph.png)

| Application Permissions | |
|-------------------------|---------------------|
| Directory | Read directory data |

How to Set up Azure Active Directory to perform authentication using Single Sign-On for Bold Reports On-Premise Steps to set up Azure Active Directory for Bold Reports On-Premise

| Delegated Permissions | |
|-----------------------|-------------------------------------------------------------|
| 1. Directory | Read directory data, Access directory as the signed in user |
| 2. User | Sign in and read user profile |

Select the above listed permissions and click on Add permissions button.

![Add Permissions] (/static/assets/on-premise/images/how-to/add-permissions.png)

- After adding the permissions, click **Grant admin consent** to grant the admin consent for these permission.

![Grant Admin Consent](/static/assets/on-premise/images/how-to/grant-admin-consent.png)

10. Select Expose an API in the left side menu and click on set from Application ID URI.

![App Id URI](/static/assets/on-premise/images/how-to/set.png)

Enter App ID URI and click on Save button

![App Id URI](/static/assets/on-premise/images/how-to/app-id-uri-save.png)

The Application ID URI must be in the format `http://{directory domain name}/{application id}`

11. Select **Expose an API** in the left side menu and click on **Add a scope** button.

![App a scope option](/static/assets/on-premise/images/how-to/add-a-scope-option.png)

Enter Scope name, choose Admins and users on consent and enter Admin consent display name, Admin consent description, User consent display name, User consent description. Choose the state as Enabled. Click on Add scope button.

![App a scope](/static/assets/on-premise/images/how-to/add-a-scope-button.png)

The Redirect URI and Home page URL should be the URL of the Bold Reports On-Premise application.

Steps to register Bold Reports On-Premise mobile application in Azure Active Directory

1. Enter into the respective directory. Click App registrations in the left side menu and then click New registration to add a new application.

![[Add Application](/static/assets/on-premise/images/how-to/add-application-1.png)]

2. Enter the name of the application and choose the following options,

[\[!Register\]\(/static/assets/on-premise/images/how-to/register-option-mob.png\)](#)

- Accounts in any organizational directory (Any Azure AD directory - Multitenant) as **Supported account types**.

How to Set up Azure Active Directory to perform authentication using Single Sign-On for Bold Reports On-Premise

Steps to set up Azure Active Directory for Bold Reports On-Premise

- Public client/native(mobile & desktop) under the **Redirect URI(optional)** section and enter the Redirect URI. And then click **Register**.

The application will be added to the directory and you can view the details of the application in the **App registrations**.

3. Select **Branding** in the left side menu and enter the **Home page URL** and click **Save**.

![Branding](/static/assets/on-premise/images/how-to/branding.png)

4. Select **Expose an API** in the left side menu and click on **set** from Application ID URI.

![App Id URI](/static/assets/on-premise/images/how-to/set.png)

Enter **App ID URI** and click on **Save** button

![App Id URI](/static/assets/on-premise/images/how-to/app-id-uri-save.png)

The **Application ID URI** must be in the format `http://{directory domain name}/{application id}`

The **Home page URL** should be the URL of the Bold Reports On-Premise application.

Configure the Azure Active directory details in Bold Reports On-Premise to perform Single Sign-On

- Configure the following fields in the Bold Reports On-Premise to perform Single Sign-On in Bold Reports On-Premise.

Application Id: Go to the registered application and click the **Overview**, and then copy the **Application Id** and paste it.

![Application ID](/static/assets/on-premise/images/how-to/client-id.png)

Application Id URI: Go to the registered application and click the **Overview**, and then copy the **Application Id URI** and paste it.

![Application ID URI](/static/assets/on-premise/images/how-to/app-id.png)

Tenant Name: It is the default domain name of your Active Directory. Go to the created **Azure Active Directory** and copy the domain name.

![Tenant name](/static/assets/on-premise/images/how-to/tenant-name.png)

Mobile App Client ID: Go to the registered application for Bold Reports On-Premise mobile application. Copy the **Application Id** and paste it.

![Mob app client ID](/static/assets/on-premise/images/how-to/mob-app-client-id.png)

Configure the Azure Active directory details in Bold Reports On-Premise to import users and groups

- Configure the following fields in Bold Reports On-Premise settings to import Azure AD users and groups.

Tenant Name: It is the default domain name of your Active Directory. Go to the created Azure Active Directory and copy the domain name.

![Tenant name](/static/assets/on-premise/images/how-to/tenant-name.png)

Client ID: It is the Client Id of the Bold Reports On-Premise application in your Azure Active Directory. Go to the registered application and then copy the Application Id in the Overview and paste it here.

![Client ID](/static/assets/on-premise/images/how-to/client-id.png)

Client secret code: It is the secure key of the Bold Reports On-Premise application you created in your Azure Active Directory. Go to the Certificates & secrets and search for the Keys you saved for the application and then choose the Value.

![Client-Secret](/static/assets/on-premise/images/how-to/client-secret-value.png)

After the settings are configured in Bold Reports On-Premise, the Azure user can be imported into the Bold Reports On-Premise. Refer to the following link to [Import Azure Active Directory Users](#) and [Import Azure Active Directory Groups](#).

How to configure Gmail SMTP Server in Email Settings

This section explains about how to configure the Gmail SMTP Server in Email Settings of Bold Reports On-Premise.

Gmail SMTP Server Details

- To send emails from your Gmail SMTP, you must allow less secure apps access in your google account.
- Follow below steps to enable less secure apps access.
 - Login to your google account.
 - Navigate to this [link](#) and enable the access.
 - The changes will not takes place as immediately. Hence, let we wait for an few minutes to update the settings by google.
- We have to save below Gmail SMTP Server details in the [email settings](#) of Bold Reports On-Premise.

SMTP Server - smtp.gmail.com

SMTP Port - 587

Sender Name - Bold Reports Server

Sender Email Address - Your Gmail Address

Authentication Type - Basic Authentication

Username - Your Gmail Address

Password - Your Gmail Password

Enable SSL - True

Anonymous Authentication is not supported in the Gmail SMTP Server.

Report link with filter parameters

You can pass the parameters to a report by including them in a report URL. Passing parameter values within URL will apply filter in the report on initial load itself.

To set a report parameter within a URL, use the following syntax.

parameter=value1, value2,..., valueN

Where **parameter** represents the parameter name.

Parameter can be single-valued and multiple-valued.

To append your query string made with parameters and values, to a URL, add a prefix (?) to the query string. If (?) is already there in the URL, add a prefix(&) to the query string.

`http://<servername>/<culturename>/reports/<reportid>/<category>/<reportname>?Paramete
rName=Value`

Here is a report view illustrating the same with single-valued parameter.

![URLFilterParameterValue](/static/assets/on-premise/images/how-to/pass-singleparameter-in-url.png)

Here is a report view illustrating the same with multi-valued parameter.

![URLFilterParameterValues](/static/assets/on-premise/images/how-to/pass-multipleparameter-in-
url.png)

How to filter data based on user in a report

In on-premise server, we can filter and display the data based on user who runs the report. In this section, we have explained the report design which displays the sales history of a salesperson who runs the report. Follow the step by step procedure to design such report in Bold Reports Designer.

First we need to connect to the data, and present it in table format using table report item.

Connecting to Data

Here to showcase the sales history, sample data is prepared by creating a temporary table using SQL query in the query designer. You can also [connect to your data](#) from any of your server or local database.

1. Click the **Data** icon in the configuration panel to launch a **Data** configuration pane.

![Report Designer page](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/data-pane.png)

2. Click on the **New Data** button in Data panel.
3. In the connection type panel, click the data source type that you want to connect. Here, **SQL** connection type is used to demonstrate.

![Connection types](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/connection-types.png)

4. In the new data source configuration panel, fill the server name and related details. As I said earlier, we are going to create a data set by temporary table concept. So, you can provide credentials and connect with any of your server or local databases.

![Connection panel](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/new-datasource.png)

5. Click on the **Connect** button, now the following view will be displayed.

![Query builder](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/query-designer.png)

6. In the query builder, you can build query with existing tables in your database or use the below query.

```
'sql
DROP TABLE UserInfo
CREATE TABLE UserInfo
(
    user_id INT IDENTITY PRIMARY KEY,
    SalesPersonID VARCHAR(200) NOT NULL,
    SalesPerson VARCHAR(100) NOT NULL,
    CustomerName VARCHAR(100) NOT NULL,
    ProductName VARCHAR(100) NOT NULL,
    PurchaseDate DATETIME NOT NULL,
    ExpiryDate DATETIME NOT NULL,
    Amount Float,
    LicenseType VARCHAR(100) NOT NULL
)
INSERT INTO UserInfo
VALUES ('smith@syncfusion.com','Smith', 'James', 'WEB (Essential JS 2)', '3/1/2016', '3/1/2017','23456', 'Standard'),
('smith@syncfusion.com','Smith', 'Jeff', 'WEB (Essential JS 1)', '9/8/2019', '9/8/2020','12323', 'Community'),
('smith@syncfusion.com','Smith', 'Armstrong', 'Embedded Reporting Tools', '1/12/2018', '1/12/2020','79897', 'Standard'),
('smith@syncfusion.com' 'Smith', 'Cromley', 'EnterpriseBI', '1/12/2018', '1/12/2020','45364', 'Platinum'),
('smith@syncfusion.com','Smith', 'Richardson', 'EnterpriseBI', '2/11/2017', '2/11/2020','21467', 'Platinum'),
```

```

('smith@syncfusion.com','Smith', 'Horton', 'Cloud BI', '9/11/2018', '2/11/2020','87897', 'Standard'),
('smith@syncfusion.com','Smith', 'Washington', 'Embedded BI', '3/8/2019', '3/8/2020','13446',
'Platinum'),
('smith@syncfusion.com','Smith', 'Victoria', 'WEB (Essential JS 2)', '3/8/2019', '3/8/2020','43566',
'Standard'),
('smith@syncfusion.com','Smith', 'Eric', 'FILE FORMAT', '3/8/2004', '8/12/2005','67899', 'Community'),
('smith@syncfusion.com','Smith', 'John', 'MOBILE', '8/8/2019', '8/8/2020','89800', 'Standard'),
('anderson@syncfusion.com','Anderson', 'Eric', 'WEB (Essential JS 2)', '1/1/2016', '1/1/2017','23456',
'Standard'),
('anderson@syncfusion.com','Anderson', 'James', 'WEB (Essential JS 1)', '10/8/2019',
'10/8/2020','12323', 'Community'),
('anderson@syncfusion.com','Anderson', 'Victoria', 'Embedded Reporting Tools', '11/12/2018',
'11/12/2020','79897', 'Standard'),
('anderson@syncfusion.com','Anderson', 'Charles', 'EnterpriseBI', '12/12/2018', '12/12/2020','45364',
'Platinum'),
('anderson@syncfusion.com','Anderson', 'Horton', 'EnterpriseBI', '5/11/2017', '5/11/2020','21467',
'Platinum'),
('anderson@syncfusion.com','Anderson', 'Richard', 'Cloud BI', '6/11/2018', '6/11/2020','87897',
'Standard'),
('anderson@syncfusion.com','Anderson', 'John', 'Embedded BI', '4/8/2019', '4/8/2020','13446',
'Platinum'),
('anderson@syncfusion.com','Anderson', 'Steffy', 'WEB (Essential JS 2)', '12/8/2019',
'12/8/2020','43566', 'Standard'),
('anderson@syncfusion.com','Anderson', 'Pat', 'FILE FORMAT', '7/8/2004', '7/12/2005','67899',
'Community'),
('anderson@syncfusion.com','Anderson', 'Moffy', 'MOBILE', '5/8/2019', '5/8/2020','89800', 'Standard')
SELECT * FROM UserInfo
`
```

Note: The above query is static sample data created for demonstration purpose.

7. Switch to the query mode and enter the query.

![Code mode](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/enter-query.png)

Create query parameter

To filter the user specific data, create a parameter at query level. Add the below WHERE clause statement in the query mode.

```
`sql
```

WHERE SalesPersonID = @userid

Filter data using user collection

The User collection provides detail about the user who ran the report. To filter the sales history, passing the sales person id as value for **user_id** parameter. Follow the below steps to assign the value for parameter.

1. Click on the Parameter icon in the query designer toolbar.

![Report parameter option in toolbar](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/parameter-icon.png)

2. By default, the Parameters dialog will be open with **@user_id** parameter.

![Parameters dialog](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/parameter-dialog.png)

3. Click on the small square icon next to the **Value** field.

![Option to add expression](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/expression-icon.png)

4. It will launch the expression dialog like below.

![Expression builder](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/expression-dialog.png)

5. Enter the **=User!UserID** expression in the text area and click OK.

![Enter expression to filter data](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/enter-expression.png)

6. Click **Finish**.

When the report runs the data set values will be filtered based on the current user who runs the report.

Display data in table

The table report item is listed in the item panel under the Data Regions category.

![Table report item in item panel](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/itempanel.png)

Drag and drop the table report item into the design area from the item panel.

![Add table to the design](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/add-table.png)

Assign the required data fields in the table cell and format the report design as required using the properties listed in property panel. Refer [Simple Table Design](#) section to learn the designing steps in detail.

The below report design will display the customer name, purchase details and sales amount achieved by a salesperson when the report runs.

![Simple table design to view sales history](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/design-view.png)

Adding header to the report

To complete the design, add a header to the report. In the report header, we can show the salesperson name and title for the report. Follow the below steps to add header to the report.

1. Right-click on the design surface and select **Add Header** from context menu.

![Add header option in context menu](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/add-report-header.png)

2. To add the text to the header, drag the **textbox** report item to the header area.

![Adding textbox in header](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/add-textbox-in-header.png)

3. Right click inside the **textbox** and select **Expression** option from context menu.

![Add expression in text box](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/expression-menu.png)

4. In the expression builder, assign the `=First(Fields!SalesPerson.Value,"DataSet1")` expression.

![Enter expression to display sales person](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/add-expression-value-in-text-box.png)

5. Click **OK**.
6. Enter **- Sales History** text in the text box.

![Enter report title](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/append-title.png)

You can format the header **textbox** using the properties listed in the properties panel.

![Final report design view](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/header-output.png)

Report Preview

Preview the report. Now based on the current user, the data will be displayed in the table as shown below.

![Sales history data of current user](/static/assets/on-premise/images/how-to/filter-data-based-on-user-in-a-report/report-preview.png)

You can download the report design from [here](#).

How to export the report from Bold Reports Report Server

You can export the report into any format(Excel, PDF, HMTL, PPT CSV, and Word) from Bold Reports Server by using the [Export Report API](#).

User must have **Read** permission for the reports.

Pass the following parameters in request body.

Report Id - Report Id.

Server Path - Specifies the relative URL of the report.

Export Type - Export Type(PDF, Excel, Html, PPT, Word, and CSV).

Access token must be passed in authorization header. To get access token, use the [Authentication API](#).

Here is a sample C# code to illustrate the approach.

Add the following assemblies in namespace

- System.Runtime.Serialization.Json
- System.IO
- Newtonsoft.Json
- System.Net
- System.Collections.Generic
- System.Text

Refer **Newtonsoft.Json** assembly from [nuget package](#)

```
'csharp
public class ApiExportReport
{
    public Guid ReportId { get; set; }
    public string ServerPath { get; set; }
    public string ExportType { get; set; }
}
public class ItemResponse
{
    public bool Status { get; set; }
    public byte[] FileContent { get; set; }
    public string StatusMessage { get; set; }
}
```

```
public string ApiStatus { get; set; }

}

public static ItemResponse ExportReport()
{

var BoldReportsURL = "https://on-premise-demo.boldreports.com";
var itemId = Guid.Parse("b401dfc7-91f8-42a6-9f30-0b3e1323a5cc");
var exporttypeId = "PDF";
var itemRequest = new ApiExportReport
{
    ReportId = itemId,
    ExportType = exporttypeId
};

using (var proxy = new Custom WebClient())
{
    var ser = new DataContractJsonSerializer(typeof(ApiExportReport));
    var mem = new MemoryStream();
    ser.WriteObject(mem, itemRequest);
    proxy.Headers["Content-type"] = "application/json";
    proxy.Headers["Authorization"] = token.tokentype + " " + token.accesstoken; // token must be passed here
    proxy.Encoding = Encoding.UTF8;
    var data = Encoding.UTF8.GetString(mem.ToArray(), 0, (int)mem.Length);
    try
    {
        var rdata = proxy.UploadString(new Uri(BoldReportsURL +
        "/reporting/api/site/site1/v1.0/reports/export"), "POST", data);
        var result = JsonConvert.DeserializeObject<ItemResponse>(rdata);
        File.WriteAllBytes("D://Test.pdf", result.FileContent);
        return result;
    }
    catch (WebException ex)
    {
        if (ex.Response is HttpWebResponse)
    }
}
```

```
var resp = new StreamReader(ex.Response.GetResponseStream()).ReadToEnd();
dynamic obj = JsonConvert.DeserializeObject(resp);
Console.WriteLine(obj);
}
}
return null;
}
}
class CustomWebClient : WebClient
{
protected override WebRequest GetWebRequest(Uri uri)
{
var request = base.GetWebRequest(uri);
request.Timeout = 4 * 60 * 1000; //Increase time out
if (request is HttpWebRequest)
{
(request as HttpWebRequest).KeepAlive = false;
}
return request;
}
}
`
```

How to generate the access token for Bold Reports Report Server user

You can generate the access token for the user in Bold Reports by using the [Authentication API](#). For this, pass the following parameters in request body.

username - Email address of the user.

password - Password of the user.

grant_type - This type of credentials are used to authorize the request for an access token. Valid values:
password.

Here is a sample C# code to illustrate the approach.

Add the following assemblies in namespace

- System.Collections.Generic
- System.Net.Http

- Newtonsoft.Json

Refer **Newtonsoft.Json** assembly from [nuget package](#)

csharp

```
public class Token
{
    public string access_token { get; set; }
    public string token_type { get; set; }
    public string expires_in { get; set; }
    public string userName { get; set; }
}

private static string tokenurl = "/reporting/api/site/site1/token";
public static Token GenerateToken(string userName, string password)
{
    try
    {
        var BoldReportsURL = "https://on-premise-demo.boldreports.com";
        using (var client = new HttpClient())
        {
            client.BaseAddress = new Uri(BoldReportsURL);
            client.DefaultRequestHeaders.Accept.Clear();
            var content = new FormUrlEncodedContent(new[]
            {
                new KeyValuePair<string, string>("grant_type", "password"),
                new KeyValuePair<string, string>("username", userName),
                new KeyValuePair<string, string>("password", password)
            });
            var result = client.PostAsync(tokenurl, content).Result;
            string resultContent = result.Content.ReadAsStringAsync().Result;
            return JsonConvert.DeserializeObject<Token>(resultContent);
        }
    }
    catch (Exception)
    {
```

```
//ignored  
}  
  
return new Token();  
}
```

After token is generated, use the generated token by attaching in the request header(Authorization) for all subsequent API calls to authenticate the requests.

How to get the list of items from Bold Reports Report Server

You can get the list of items(Category, Reports, Dataset, Data source, and Schedule) from Bold Reports by using the [Get Items API](#).

For this, pass the following parameters in query parameters.

itemType - Pass the item type(Category, Reports, Dataset, Data source, and Schedule).

serverPath - Category Path.

Access token must be passed in authorization header. To get access token, use the [Authentication API](#).

Here is a sample C# code to illustrate the approach.

Add the following assemblies in namespace

- System.IO
- Newtonsoft.Json
- System.Net
- System.Collections.Generic
- System.Text

Refer **Newtonsoft.Json** assembly from [nuget package](#)

```
'csharp  
public class Apiltems  
{  
    public bool CanRead { get; set; }  
    public bool CanWrite { get; set; }  
    public bool CanDelete { get; set; }  
    public bool CanSchedule { get; set; }  
    public bool CanDownload { get; set; }  
    public bool CanOpen { get; set; }  
    public bool CanMove { get; set; }  
    public bool CanCopy { get; set; }}
```

```
public bool CanClone { get; set; }

public bool CanCreateItem { get; set; }

public Guid? CategoryId { get; set; }

public string CategoryName { get; set; }

public int ModifiedById { get; set; }

public string CreatedByDisplayName { get; set; }

public int CreatedById { get; set; }

public string ModifiedByFullName { get; set; }

public string ItemLocation { get; set; }

public int ItemType { get; set; }

public Guid Id { get; set; }

public string CreatedDate { get; set; }

public string ModifiedDate { get; set; }

public DateTime ItemModifiedDate { get; set; }

public DateTime ItemCreatedDate { get; set; }

public Guid ReportId { get; set; }

public string ReportName { get; set; }

public string Name { get; set; }

public string Description { get; set; }

}

public static List<Apiltems> GetItems()

{

var BoldReportsURL = "https://on-premise-demo.boldreports.com";

var itemType = "Report";

using (var proxy = new Custom WebClient())

{

proxy.Headers["Content-type"] = "application/json";

proxy.Headers["Authorization"] = token.tokentype + " " + token.accesstoken; // token must be passed here

proxy.Encoding = Encoding.UTF8;

try

{



var rdata = proxy.DownloadString(new Uri(BoldReportsURL + "/reporting/api/site/site1/v1.0/items?itemType=" + itemType));
```

```
var response = JsonConvert.DeserializeObject<List<Apiltems>>(rdata);
return response;
}
catch (WebException ex)
{
if (ex.Response is HttpWebResponse)
{
var resp = new StreamReader(ex.Response.GetResponseStream()).ReadToEnd();
dynamic obj = JsonConvert.DeserializeObject(resp);
Console.WriteLine(obj);
}
}
return null;
}
}

class CustomWebClient : WebClient
{
protected override WebRequest GetWebRequest(Uri uri)
{
var request = base.GetWebRequest(uri);
request.Timeout = 4 * 60 * 1000; //Increase time out
if (request is HttpWebRequest)
{
(request as HttpWebRequest).KeepAlive = false;
}
return request;
}
}
`
```

How to apply the patches to Bold Reports Multi-tenant Azure App Service

The support team provides solutions to reported issues in the form of patch files and this topic explains about how to apply these patch files to the Bold Reports Multi-tenant Azure App Service.

Steps to apply patches in Bold Reports Multi-tenant Azure App Service.

1. Log in to the [Azure portal](#).
2. Select the Bold Reports Azure App Service.
3. Stop the Bold Reports Azure App Service, and then click the Get publish profile.
4. Save the , PublishSettings file and open it. The file contains three sections for Web Deploy, FTP, and ReadOnly-FTP.
5. From the FTP section, copy the following values:

publishUrl

userName

userPWD

![publishProfile](/static/assets/on-premise/images/how-to/publish-profile.png)

6. Download and install the FileZilla client application.
7. After installing the application, set the values obtained from the PublishSettings file accordingly:

Host: Give the publishUrl value.

Username: Give the userName value.

Password: Give the userPWD value.

8. After entering these values as shown in the following image, click Quickconnect.

![Connect With Filezilla](/static/assets/on-premise/images/how-to/connecting-values-in-file-zilla.png)

9. Download and extract the files provided in the support incident.
10. Move the folders to the hosted location of the Report Server /site/wwwroot/Report Server/ as follows.

![Report Server Location](/static/assets/on-premise/images/how-to/report-server-location.png)

If the patch was provided for IDP, then move the folders to the hosted location of the IDP /site/wwwroot/IDP/.

![IDP Location](/static/assets/on-premise/images/how-to/idp-location.png)

11. Start the Bold Reports Azure App Service.

How to switch the installed Bold Reports Report Server IIS Express application to IIS

By using the DeployIIS utility, you can switch the installed Bold Reports Report Server IIS Express application to IIS and to do that follow these steps:

How to grant access to all users for Report Server

Grant access to all users in new application

1. Run the **DeployIIS** application from the following installed location to host the Bold Reports Report Server in IIS.

installed_Location\Bold Reports\On-Premise Edition\Utilities\DeployIIS\DeployIIS

2. Type a unused port for the Bold Reports Report Server and click the **Submit** as shown in the following image.

![Deploy IIS](/static/assets/on-premise/images/how-to/deployiis.png)

3. It will host the application in IIS and now you can open the Bold Reports Report Server from IIS by clicking the browse.

![Bold Reports site in IIS](/static/assets/on-premise/images/how-to/bold-reports-in-iis.png)

4. If Bold Reports Report Server [application](#) was configured already, then follow these steps to update the new binding in Bold Reports Report Server else ignore the following steps:

a. Now, navigate to the site settings page of the UMS application using the old IIS Express binding URL and update the new binding information as shown in the following image.

<http://localhost:{port-no}/ums/administration>

![UMS Site Setting](/static/assets/on-premise/images/how-to/ums-site-settings.png)

By default, Bold Reports Report Server will be opened with old IIS Express binding URL.

b. Then, navigate to the site settings of your Report Server application using the old IIS Express binding URL and update the new binding information as shown in the following image.

<http://localhost:{port-no}/reporting/en-us/site/site1/administration>

![Report Server Site Setting](/static/assets/on-premise/images/how-to/report-server-site-settings.png)

c. Now, Bold Reports Report Server site will be updated with new bindings and you can start Bold Reports Report Server site with new bindings.

How to grant access to all users for Report Server

Grant access to all users in new application

1. Go to **Application Management** section in Bold User Management Server.

![Manage Application in User Management Server](/static/assets/on-premise/images/faq/manage-application-in-ums.png)

2. Click on **Add Application** button in the **Application Management** section.

![Add application in User Management Server](/static/assets/on-premise/images/faq/add-application-in-ums.png)

Where can i find the error and debug log files

Grant access to all users in the existing application

3. In the **Add Application** dialog, enable the **Allow access to all users** option to grant access to all users.

![Grant Access to all users for New Application in User Management Server](/static/assets/on-premise/images/faq/allow-access-to-all-users-in-ums.png)

Grant access to all users in the existing application

1. Go to **Application Management** and then **Choose your application**.
2. Click on **edit** option in the context menu of **Actions** button.

![Grant Access to all users in Existing Application](/static/assets/on-premise/images/faq/enable-allow-access-to-all-users-in-existing-application-of-ums.png)

3. In the **Edit Application** dialog, enable the **Allow access to all users** option.

![Enable grant access to all users in UMS](/static/assets/on-premise/images/faq/edit-existing-application-in-ums.png)

4. Click on **Update** button.

Where can i find the error and debug log files

Error logs

Error log files are generated when an exception occurs while configuring or interacting with the Bold Reports application

Debug logs

Event log files record the complete user interaction details one after the other when users interact with the Bold Reports application

Log Directories

For any configurations made in the Bold Reports, log files are generated in deployed locations under the various modules listed in the following table based on the nature of the error or event.

By default, the Bold Reports is deployed in C:\Bold Reports.

| Application | Log location |
|-------------------|-------------------------------------------------|
| Identity Provider | {Deployed Location}\IDP\App_Data\Logs |
| Report Server | {Deployed Location}\Report Server\App_Data\Logs |

What are the features need to be enabled in IIS to run the Bold Reports Application in Windows Client OS

Bold Reports On-Premise Edition can be hosted in an IIS express and IIS. To run the Bold Reports in IIS, enable the IIS along with IIS features and roles.

What are the features need to be enabled in IIS to run the Bold Reports Application in Windows Client OS Steps to enable the IIS and features that are needed to run the Bold Report Server in Windows Client OS

Steps to enable the IIS and features that are needed to run the Bold Report Server in Windows Client OS

1. Open Control Panel and click Programs and Features > Turn Windows features on or off.
2. Enable Internet Information Services.

![Control Panel](/static/assets/on-premise/images/faq/windows-features.png)

3. Expand the .NET Framework 4.5 Advanced Services and enable the ASP.NET 4.5 feature.

You should use Microsoft .NET Framework 4.5 or higher version. Learn more [here](#).

![Roles and Features](/static/assets/on-premise/images/faq/ms-framework.png)

4. Expand the Internet Information Services feature and verify that the required IIS component listed [here](#) are selected. Click OK.

![Roles and Features](/static/assets/on-premise/images/faq/iis-features-client.png)

Required web server components

The following listed IIS components satisfy the minimum requirements to run the Bold Reports. If other IIS components are enabled, they do not need to be removed.

| Section | IIS Components |
|-------------------------|---------------------------------------------------------------------------------------------------------------|
| World Wide Web Services | Common HTTP Features - Default Document - Directory Browsing - HTTP Errors - Static Content |
| | Health and Diagnostics - HTTP Logging |
| | Performance Features - Static Content Compression |
| | Security - Request Filtering - Windows Authentication |
| | Application Development - .NET Extensibility 4.8 - ASP.NET 4.8 - ISAPI Extensions - ISAPI Filters |
| Web Management Tools | IIS Management Console |

What are the features need to be enabled in IIS to run Bold Reports Application in Windows Server OS

Steps to enable the IIS and features that are needed to run the Bold Reports Server in Windows Server OS

What are the features need to be enabled in IIS to run Bold Reports Application in Windows Server OS

Bold Reports On-Premise Edition can be hosted in an IIS express and IIS. To run the Bold Reports in IIS, enable the IIS along with IIS features and roles.

Steps to enable the IIS and features that are needed to run the Bold Reports Server in Windows Server OS

1. Open Server Manager and click Manage > Add Roles and Features. Click Next.
2. Select Role-based or feature-based installation and click Next.
3. Choose Select a server from the server pool and select the server in the Server Pool section, and then click Next.
4. On Server Roles, enable the Web Server (IIS) in the Roles section.
5. On Features, enable the following .NET Framework features and click Next.

You should use Microsoft .NET Framework 4.5 or higher version. Learn more [here](#).

![Roles and Features](/static/assets/on-premise/images/faq/roles-features.png)

6. On the Web Server Role (IIS) dialog box, click Next.
7. On the Select role services dialog box and verify that the required web server component listed [here](#) are enabled. Click Next.

![Roles Services](/static/assets/on-premise/images/faq/role-services.png)

8. Verify that your settings are correct and click Install.
9. When the installation completes, click Close to exit the wizard.

Required web server components

The following listed IIS components satisfy the minimum requirements to run the Bold Reports. If other IIS components are enabled, they do not need to be removed.

| Section | IIS Components |
|------------|---------------------------------------------------------------------------------------------------------|
| Web Server | Common HTTP Features - Default Document - Directory Browsing - HTTP Errors - Static Content |
| | Health and Diagnostics - HTTP Logging |
| | Performance Features - Static Content Compression |

| Section | IIS Components |
|------------------|---------------------------------------------------------------------------------------------------------------|
| | Security - Request Filtering - Windows Authentication |
| | Application Development - .NET Extensibility 4.6 - ASP.NET 4.6 - ISAPI Extensions - ISAPI Filters |
| Management Tools | IIS Management Console |

Centralized report authoring

Centralized report authoring explains how to use the report with authors and multiple developers with several access.

Is Bold Reports have a centralized report authoring system

Yes, Bold Reports having On-Premise and Cloud version Report Server to have the reports in centralized report authoring.

We can refer more details of this from [Bold Report Cloud](#) and [Bold Reports On-Premise](#).

Will Syncfusion charge for Bold Reports Azure App Service

No, Syncfusion will not charge for Bold Reports Azure App Service. But, you should have an active/trial Enterprise or Embedded subscriptions with Bold Reports for using our Bold Reports Report Server with Azure App service.

![Bold Report accounts](/static/assets/on-premise/images/faq/bold-reports-account.png)

Where can i find the error and debug log files of Bold Reports Azure App Service

Error logs

Error log files are generated when an exception occurs while configuring or interacting with the Bold Reports application.

Debug logs

Event log files record the complete user interaction details one after the other when users interact with the Bold Reports application.

Log directories

For any configurations made in the Bold Reports, log files are generated in deployed locations under the various modules listed in the following table based on the nature of the error or event.

By default, the Bold Reports Azure App Service will be deployed in /site/wwwroot/ location.

| | |
|-------------|--------------|
| Application | Log Location |
|-------------|--------------|

| | | |
|-------------------|-------------------------------------------------|--|
| ----- ----- | | |
| Identity Provider | {Deployed Location}\IDP\App_Data\Logs | |
| Report Server | {Deployed Location}\Report Server\App_Data\Logs | |

Steps to get the log files from Bold Reports Azure App service

1. Log in to the [Azure portal](#).
2. Select the Bold Reports Azure App Service.
3. Stop the Bold Reports Azure App Service, and then click the Get publish profile.
4. Save the , PublishSettings file and open it. The file contains three sections for Web Deploy, FTP, and ReadOnly-FTP.
5. From the FTP section, copy the following values:

publishUrl

userName

userPWD

![publishProfile](/static/assets/on-premise/images/how-to/publish-profile.png)

6. Download and install the FileZilla client application.
7. After installing the application, set the values obtained from the PublishSettings file accordingly:

Host: Give the publishUrl value.

Username: Give the userName value.

Password: Give the userPWD value.

8. After entering these values as shown in the following image, click **Quickconnect**.

![Connect With Filezilla](/static/assets/on-premise/images/how-to/connecting-values-in-file-zilla.png)

9. You can find the logs of Report Server and IDP application from the following location as shown in the table.

| | | |
|------------------------------------------------------------------------------------------------------------------------------------|--|--|
| Application Log location Log location image | | |
| ----- ----- | | |
| /site/wwwroot/IDP/App_Data/Logs ![Report Server Logs Location](/static/assets/on-premise/images/how-to/idp-logs.png) | | |
| /site/wwwroot/Report Server/App_Data/Logs ! [IDP Logs Location] (/static/assets/on-premise/images/how-to/report-server-logs.png) | | |

Why should you migrate to Bold Reports from Syncfusion Report Platform

As per our plan to have a separate suite for the reporting platform, we have introduced Bold Reports from Syncfusion. Due to the migration to Bold reports, there will be no further feature updates for Syncfusion Report Platform. So, you need to consider using our Bold Reports for your developments.

Is it possible to migrate from Syncfusion Report Platform Report Server to Bold Reports Report Server

Yes, it is possible to migrate from Syncfusion Report Platform Report Server to Bold Reports Report Server by using our Data Migration utility and this migration will be supported with Syncfusion Report Server version from 4.x.

You can also migrate Bold Reports v1.x to Bold Reports v2.x.

See also

[Migrating Syncfusion Report Server v4.x or Bold Reports v1.x to Bold Reports v2.x](#)

[Migrating Syncfusion Report Server Azure App Service v4.x to Bold Reports Azure App Service v2.x](#)

[Migrating Bold Reports Azure App Service v1.x to Bold Reports Azure App Service v2.x](#)

Where should I replace the Bold Reports license key in Bold Reports Azure App Service

By default, the Bold Reports Azure App Service will have an expired license key. You need to replace your Bold Reports license key in the Bold Reports Azure App Service by following these steps:

1. Login into your [Azure Portal](#).
2. Press the Stop option to stop the Bold Reports Azure App Service.
3. Choose the [App Service editor](#) under the Development tool in the options as follows.

![App Service Editor](/static/assets/on-premise/images/faq/app-service-editor.png)

4. Now, the deployed files will be displayed.
5. Navigate to this location [/site/wwwroot/Report Server/Configuration/BoldLicense.txt](#) and replace your license key.
6. Then, start the Bold Reports Azure App Service.

See also

[Create Bold Reports Azure App Service using ARM template](#)

[Upgrade Bold Reports Azure App Service from v1.x to latest](#)

[Upgrade Syncfusion Report Server Azure App Service from v4.x to latest](#)

Where does the data resides in Bold Reports

Bold Reports data will be maintained the following details in the database, which was provided for configuring the [Bold Reports application](#).

- Information of reports
- Information of shared data sources
- Information of shared data sets
- Categories
- Schedules definitions
- Permissions settings associated for report, data source, data set, categories, and schedules
- Report Server configurations

The reports, dataset, and data source, which are added to the Bold Reports Server will be maintained in some location based on storage type you have chosen while configuring the Bold Reports Server application.

For **File Storage** type, reports, dataset, and data source will be maintained in the following location.

![Resources location](/static/assets/on-premise/images/faq/resource-location.png)

For **Azure Blob Storage** type, reports, datasets, and data source will be maintained in the **Azure Blob Container**, which was provided for configuring Bold Reports Server application.

See also

[Does Report Server store reports execution data to server database](#)

Does Report Server store reports execution data to the server database

Bold Reports Report Server stores the following information only to the server database. It does not store the reports execution data to the server database.

- Information of reports
- Information of shared data sources
- Information of shared data sets
- Categories
- Schedules definitions
- Permissions settings associated for report, data source, data set, categories, and schedules
- Report Server configurations

See also

[Where does the data resides in Bold Reports](#)

REST API

Using the Bold Reports On-Premise REST API, you can manage and change Bold Reports resources programmatically via HTTP. The API gives you simple access to the functionality behind the resources on a Bold Reports. You can use this access to create your own custom applications or to script interactions with Bold Reports resources.

Overview

The Bold Reports provides rich set of Embedded Reporting Tools Report Viewer, Report Designer, Report Writer and Report Server in a single point of access to design, preview, manage, schedule, export and print the reports.

Key features

[Design and Edit Reports](#) --- Report Designer allows you to create and edit the reports, datasets, various types of datasources directly in web.

[Report management](#) --- Reports can be efficiently organized under the categories. Permission to view the reports can be given to specific users or groups.

[Display Reports](#) --- You can preview the new or already created reports to take a look on generated report within your applications (Javascript, ASP.NET CORE, Angular, ASP.NET, ASP.NET MVC, WPF, UWP) using Report Viewer.

[Export Reports](#) --- You can export the RDL / RDLC report to popular formats like PDF, Excel, CSV, PowerPoint, Word and HTML using Report Writer library. Also you can export your report from Report Designer, Report Viewer and Report Server.

[Interactive Reports](#) --- End users can able to create highly customized report. You can alter the report layout in preview using filters, highlighting and sorting the data.

Overview

The Embedded Reporting Tools provides rich set of reporting tools Report Viewer, Report Designer, Report Writer and Report Server in a single point of access to design, preview, manage, schedule, export and print the reports.

Key features

[Design and Edit Reports](#) --- Report Designer allows you to create and edit the reports, datasets, various types of datasources directly in web.

[Report management](#) --- Reports can be efficiently organized under the categories. Permission to view the reports can be given to specific users or groups.

[Display Reports](#) --- You can preview the new or already created reports to take a look on generated report within your applications (Javascript, ASP.NET CORE, Angular, ASP.NET, ASP.NET MVC, WPF, UWP) using Report Viewer.

[Export Reports](#) --- You can export the RDL / RDLC report to popular formats like PDF, Excel, CSV, PowerPoint, Word and HTML using Report Writer library. Also you can export your report from Report Designer, Report Viewer and Report Server.

[Interactive Reports](#) --- End users can able to create highly customized report. You can alter the report layout in preview using filters, highlighting and sorting the data.

System Requirements

This topic describes the software and hardware requirements to run Embedded Reporting Tools.

Hardware Requirements

The following hardware requirements are necessary to run the Embedded Reporting Tools:

- 2.4 GHZ or faster, 32 bit or 64 bit processor.
- 8 GB RAM for 32 bit or 2 GB RAM for 64 bit.
- 1.68 GB Hard Disk space (Installation files).

Software Requirements

The following software requirements are necessary to run the Embedded Reporting Tools:

- Microsoft Visual Studio 2010 or later
- Internet Information Services (IIS) 7.0+

Supported Operating Systems

- Windows 7+
- Windows Server 2008 R2+

Browser Compatibility

- IE 9+
- Microsoft Edge
- Mozilla Firefox 22+
- Chrome 17+
- Opera 12+
- Safari 5+

See Also

- [Licensing procedure for deployment](#)

Download and installation

This section briefly describes steps involved in download and installation of the Embedded Reporting Tools setup.

Register and download Embedded Reporting Tools

1. Go to Embedded Reporting Tools [download](#) page.
2. Choose your plan and click on Start Your Trial option.

![Embedded Reporting Tools Installer Extraction](/static/assets/embedded-reporting-tools/images/installation/embeded-plans.png)

3. Fill the form input details to register new account or sign in with existing account.

![Embedded Reporting Tools Installer Extraction](/static/assets/embedded-reporting-tools/images/installation/registration.png)

4. Once portal have been created, you will be redirected to your downloads page.

![Embedded Reporting Tools Installer Extraction](/static/assets/embedded-reporting-tools/images/installation/download-option.png)

5. Click on download button and then select the file type to download.

![Embedded Reporting Tools Installer Extraction](/static/assets/embedded-reporting-tools/images/installation/downloan-setup.png)

Installing Embedded Reporting Tools

1. Once setup downloaded, run the installer from the saved location by clicking the **Run** button or by double-clicking the EXE file.

![Embedded Reporting Tools Installer Extraction](/static/assets/embedded-reporting-tools/images/installation/setup-extraction.png)

2. Sign-in with your registered e-mail address to unlock the setup.

![Installation Using Product Key](/static/assets/embedded-reporting-tools/images/installation/product-key-page.png)

3. Accept the License Agreement of Embedded Reporting Tools, by clicking the **License Terms and Conditions**.
4. Choose the location where you would like to install the Embedded Reporting Tools setup and samples, then choose the additional settings like start menu and desktop shortcuts creation. If you want to perform the additional tasks you can check the options, otherwise you can uncheck it and click **Install**.

![Path Selection Wizard](/static/assets/embedded-reporting-tools/images/installation/path-selection-page.png)

5. Now the installation begins. You can cancel the installation anytime by pressing **Cancel**, if you prefer.

![Progress bar Wizard](/static/assets/embedded-reporting-tools/images/installation/installation-progress.png)

6. On successful installation, the blow screen appears. Click the **Launch Control Panel** option to run Sample browser or Explore Samples.

![Finish Wizard](/static/assets/embedded-reporting-tools/images/installation/installation-finish-wizard.png)

Embedded Reporting Tools samples and demos

Embedded Reporting Tools provides control panel to access samples and demos of Reporting components for the JavaScript, Angular, ASP.NET CORE, ASP.NET, ASP.NET MVC, JSP, PHP, WPF and UWP platforms. To launch the Reporting Tools control panel, click the Desktop Shortcut or Start Menu -> Embedded Reporting Tools.

![Embedded Reporting Tools control panel](/static/assets/embedded-reporting-tools/images/samples/sdk-control-panel.png)

[View the product demos](#)

You can access the Embedded Reporting Tools demos using the following steps.

1. In the Embedded Reporting Tools control panel, choose the desired platform (JavaScript, Angular etc.).
2. Click any of the following options to know more about the selected platform.
 - Run Local Demos - To run the locally installed samples.
 - Explore Demos - Opens the local installed location.
 - What's New - To view the What's New content.
 - Release Notes - To view the Release Notes content.
 - Read Me -To view the Read Me content.
 - User Guide - To view the user guide documentation for the respective platform product.

JavaScript

To preview the JavaScript demos, follow these steps:

1. Choose JavaScript platform in Embedded Reporting Tools control panel.

![Control panel of JavaScript samples](/static/assets/embedded-reporting-tools/images/samples/javascript-control-panel-page.png)

2. Click Run Local Demos to view the locally installed demos.
3. You can explore the demo sample application from the following installed location by clicking Explore Demos option.

%localappdata%\Bold Reports\Reports Tools\Samples\JavaScript

4. You can view the online demos from [here](#).

Angular

To preview the Angular demos, follow these steps:

1. Choose Angular platform in Embedded Reporting Tools control panel.

![Control panel of Angular samples](/static/assets/embedded-reporting-tools/images/samples/angular-control-panel-page.png)

2. Click Run Local Demos to view the locally installed demos.
3. You can explore the demo sample application from the following installed location by clicking Explore Demos option.

%localappdata%\Bold Reports\Reports Tools\Samples\Angular

4. You can view the online demos from [here](#).

ASP.NET CORE

To preview the ASP.NET Core demos, follow these steps:

1. Choose ASP.NET Core platform in Embedded Reporting Tools control panel.

![Control panel of ASP.NET Core samples](/static/assets/embedded-reporting-tools/images/samples/asp-net-core-control-panel-page.png)

2. Click Run Local Demos to view the locally installed demos.
3. You can explore the demo sample application from the following installed location by clicking Explore Demos option.

%localappdata%\Bold Reports\Reports Tools\Samples\ASP.NET Core

4. You can view the online demos from [here](#).

ASP.NET MVC

To preview the ASP.NET MVC demos, follow these steps:

1. Choose ASP.NET MVC platform in Embedded Reporting Tools control panel.

![Control panel of ASP.NET MVC samples](/static/assets/embedded-reporting-tools/images/samples/asp-net-mvc-control-panel-page.png)

2. Click Run Local Demos to view the locally installed demos.
3. You can explore the demo sample application from the following installed location by clicking Explore Demos option.

%localappdata%\Bold Reports\Reports Tools\Samples\ASP.NET MVC

4. You can view the online demos from [here](#).

ASP.NET Web Forms

To preview the ASP.NET Web Forms demos, follow these steps:

1. Choose ASP.NET Web Forms platform in Embedded Reporting Tools control panel.

![Control panel of ASP.NET Web Forms samples](/static/assets/embedded-reporting-tools/images/samples/asp-net-web-forms-control-panel-page.png)

2. Click Run Local Demos to view the locally installed demos.
3. You can explore the demo sample application from the following installed location by clicking Explore Demos option.

%localappdata%\Bold Reports\Reports Tools\Samples\ASP.NET

4. You can view the online demos from [here](#).

WPF

To preview the WPF demos, follow these steps:

1. Choose WPF platform in Embedded Reporting Tools control panel.

![Control panel of WPF samples](/static/assets/embedded-reporting-tools/images/samples/wpf-control-panel-page.png)

2. You can explore the demo sample application from the following installed location by clicking **Explore Demos** option to preview the WPF reporting demos.

%localappdata%\Bold Reports\Reports Tools\Samples\WPF

UWP

To preview the UWP demos, follow these steps:

1. Choose UWP platform in Embedded Reporting Tools control panel.

![Control panel of UWP samples](/static/assets/embedded-reporting-tools/images/samples/uwp-control-panel-page.png)

2. You can explore the demo sample application from the following installed location by clicking **Explore Demos** option to preview the UWP reporting demos.

%localappdata%\Bold Reports\Reports Tools\Samples\UWP

Embedded Reporting Tools nuget packages and assemblies

Embedded Reporting Tools provides nuget packages and assemblies to embed reporting functionalities into your JavaScript, Angular, React, ASP.NET CORE, ASP.NET, ASP.NET MVC, WPF and UWP platform applications.

Refer to the following Embedded Reporting Tools location to add assemblies and nuget packages into your application without using online nuget package.

Embedded Reporting Tools Nuget packages Location:

C:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages

Embedded Reporting Tools Assemblies location:

C:\Program Files (x86)\Bold Reports\Reporting Tools\Assemblies

Script References

Report viewer themes and script files available from the Embedded Reporting Tools build installation location, C:\Program Files (x86)\Bold Reports\Reporting Tools\Javascript\assets.

JavaScript Embedded Reporting Tools

The following table JavaScript Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|-------------------------------|-----------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Client Side Script and Styles | <u>BoldReports.JavaScript</u> | BoldReports.JavaScript package contain Report Viewer and Report Designer HTML5 and JavaScript components for web development. |
| Server Side Helper | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

Angular Embedded Reporting Tools

The following table Angular Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|--------------------|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| Server Side Helper | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

ASP.NET Core Embedded Reporting Tools

The following table ASP.NET Core Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|------------------------|------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side Tag Helper | <u>BoldReports.AspNet.Core</u> | BoldReports.AspNet.Core package contain runtime assemblies for building line-of-business applications that can run on Windows, Linux, and Mac. It contains HTML Helpers for Report Viewer and Report Designer components to build server-side dynamic websites based on Microsoft's ASP.NET Core. |
| Server Side Helper | <u>BoldReports.Net.Core</u> | BoldReports.Net.Core is a server-side helper package to build ASP.NET Core Web API service that is used for Report Viewer and Report Designer controls with ASP.NET Core, Angular, and JavaScript platforms. |

Assemblies Details

Assembly location: C:\Program Files (x86)\Bold Reports\Reporting Tools\Assemblies\aspnetcore.

ASP.NET MVC Embedded Reporting Tools

The following table ASP.NET MVC Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|---------|----------------------|-------------|
| | | |

Embedded Reporting Tools nuget packages and assemblies ASP.NET WebForms Embedded Reporting Tools

| | | |
|------------------------|------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side MVC Helper | <u>BoldReports.Mvc4</u> <u>BoldReports.Mvc5</u> | BoldReports.Mvc4, and BoldReports.Mvc5 packages contains HTML Helpers for Report Viewer and Report Designer components to build server-side dynamic websites based on Microsoft's ASP.NET MVC. |
| Server Side Helper | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

ASP.NET WebForms Embedded Reporting Tools

The following table ASP.NET WebForms Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|------------------------|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side Tag Helper | <u>BoldReports.Webforms</u> | The BoldReports.WebForms package contains HTML Helpers for Report Viewer and Report Designer components to build server-side dynamic websites based on Microsoft's ASP.NET Web Forms. |
| Server Side Helper | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

WPF Embedded Reporting Tools

The following table WPF Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|---------------------|----------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Components Packages | <u>BoldReports.Wpf</u> | BoldReports.Wpf is a .NET Report Viewer, Report Writer control package to view and export RDL/RDLC reports within a .NET application. |

UWP Embedded Reporting Tools

The following table UWP Embedded Reporting Tools nuget packages and assemblies details.

| Purpose | Bold Reports Package | Description |
|---------------------|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Components Packages | <u>BoldReports.UWP</u> | BoldReports.UWP is a Report Viewer control package to display RDL/RDLC reports within the Universal Windows Platform application. It builds the server-side implementations for Report Viewer component. |
| Server Side Helper | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

NPM Packages

The Embedded Reporting Tools components NPM package details are tabulated below:

| Platform | Bold Reports Package |
|------------|-----------------------------------|
| Javascript | BoldReports.JavaScript |
| Extensions | BoldReports.JavaScript.Extensions |
| Globalize | BoldReports.Global |

The above Javascript NPM packages contain the following scripts and styles required for Reporting components.

| Reporting Component | Styles | Scripts | Usage |
|---------------------|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Report Viewer | bold.reports.all.min.css | bold.reports.common.min.js bold.reports.widgets.min.js ej.chart.min.js ej2-base.min.js ej2-data.min.js ej2-pdf-export.min.js ej2-svg-base.min.js ej2-lineargauge.min.js ej2-circulargauge.min.js ej.map.min.js bold.report-viewer.min.js | bold.reports.all.min.css - It contains the styles and css references of reporting dependent components. bold.reports.common.min.js - Common script for reporting widgets. bold.reports.widgets.min.js - It contains the scripts of dependent Syncfusion controls that are common for both Report Designer and Report Viewer. ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart report item. ej2-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item. ej2-data.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item. |

| | | | |
|-----------------|-----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p><code>ej2-pdf-export.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-svg-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-lineargauge.min.js</code> - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p><code>ej2-circulargauge.min.js</code> - Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Renders the Syncfusion Report Viewer widget.</p> |
| Report Designer | <code>bold.reports.all.min.css</code> <code>bold.reportdesigner.min.css</code> | <code>bold.reports.common.min.js</code> <code>bold.reports.widgets.min.js</code> <code>bold.report-designer-widgets.min.js</code> <code>ej.chart.min.js</code> <code>ej2-base.min.js</code> | <code>bold.reports.all.min.css</code> - It contains the styles and css references of reporting dependent components. <code>bold.reportdesigner.min.css</code> - It contains the styles and css references of Report Designer Component. <code>bold.reports.common.min.js</code> - Common script for |

| | | |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | <p><code>ej2-data.min.js</code></p> <p><code>ej2-pdf-export.min.js</code></p> <p><code>ej2-svg-base.min.js</code></p> <p><code>ej2-lineargauge.min.js</code></p> <p><code>ej2-circulargauge.min.js</code></p> <p><code>ej.map.min.js</code></p> <p><code>bold.report-viewer.min.js</code></p> <p><code>bold.report-designer.min.js</code></p> | <p>reporting widgets.</p> <p><code>bold.reports.widgets.min.js</code> - It contains the scripts of dependent controls that are common for both Report Designer and Report Viewer.</p> <p><code>bold.report-designer-widgets.min.js</code> - It contains the scripts of Report Designer dependent controls.</p> <p><code>ej.chart.min.js</code> - Renders the chart item. Add this script only if your report contains the chart report item.</p> <p><code>ej2-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-data.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-pdf-export.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-svg-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-lineargauge.min.js</code> - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p><code>ej2-circulargauge.min.js</code> -</p> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| | | | |
|--|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Previews the reports designed with Report Designer.</p> <p><code>bold.report-designer.min.js</code> - Renders the Syncfusion Report Designer widget.</p> |
|--|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Migrate Reporting Application

This section provides step-by-step instructions for migrating Embedded Reporting Tools such as Report Viewer, Report Designer and Report Writer from Syncfusion Essential Studio release version to Bold Reports version:

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Migrate JavaScript Embedded Reporting Tools

The following provides step-by-step instructions for migrating your JavaScript reporting application.

| Purpose | Essential Studio Packages | Bold Reports Package | Description |
|-------------------------------|-----------------------------------------------------------------------------------------|-----------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side Script and Styles | <code>Syncfusion.JavaScript.ReportDesigner</code> <code>Syncfusion.JavaScript</code> | <u>BoldReports.JavaScript</u> | <code>BoldReports.JavaScript</code> package contain Report Viewer and Report Designer HTML5 and JavaScript components for web development. |
| Server Side Helper | <code>Syncfusion.Web.ReportDesigner</code> <code>Syncfusion.Web.ReportViewer</code> | <u>BoldReports.Web</u> | <code>BoldReports.Web</code> is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

Refer the JavaScript Embedded Reporting Tools migration steps [here](#).

Migrate Angular Embedded Reporting Tools

The following provides step-by-step instructions for migrating your Angular reporting application.

| Essential Studio Packages | Bold Reports Package | Description |
|---------------------------|-------------------------------------------------------------------|------------------------------------------------------------------------------|
| syncfusion-javascript | <u>@boldreports/javascript-reporting-controls</u> | JavaScript NPM packages contain the Reporting components scripts and styles. |
| ej-angular2 | <u>@boldreports/angular-reporting-components</u> | Angular NPM packages contain the Reporting components scripts and styles. |
| syncfusion-ej-global | <u>@boldreports/global</u> | Globalize NPM packages contain the Reporting components globalize scripts. |

| Purpose | Essential Studio Packages | Bold Reports Package | Description |
|--------------------|--------------------------------------------------------------|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| Server Side Helper | Syncfusion.Web.ReportDesigner Syncfusion.Web.ReportViewer | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

Refer the Angular Embedded Reporting Tools migration steps [here](#).

Migrate ASP.NET Core Embedded Reporting Tools

The following provides step-by-step instructions for migrating your ASP.NET Core reporting application.

| Purpose | Essential Studio Packages | Bold Reports Package | Description |
|------------------------|---------------------------|-----------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side Tag Helper | Syncfusion.EJ.AspNet.Core | <u>BoldReports.AspNetCore</u> | BoldReports.AspNetCore package contains runtime assemblies for building line-of-business applications that can run on Windows, Linux, and Mac. It contains HTML Helpers for Report Viewer and Report |

| | | | |
|--------------------|------------------------------------------------------------------------------------------------------------------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | Designer components to build server-side dynamic websites based on Microsoft's ASP.NET Core. |
| Server Side Helper | Syncfusion.EJ.ReportDesigner.AspNet.Core Syncfusion.EJ.ReportViewer.AspNet.Core Syncfusion.Report.Net.Core | <u>BoldReports.Net.Core</u> | BoldReports.Net.Core is a server-side helper package to build ASP.NET Core Web API service that is used for Report Viewer and Report Designer controls with ASP.NET Core. |

Refer the ASP.NET Core Embedded Reporting Tools migration steps [here](#).

Migrate ASP.NET MVC Embedded Reporting Tools

The following provides step-by-step instructions for migrating your ASP.NET MVC reporting application.

| Purpose | Essential Studio Packages | Bold Reports Package | Description |
|------------------------|----------------------------------------------------------------------------|----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side MVC Helper | Syncfusion.AspNet.Mvc4 Syncfusion.AspNet.Mvc5 | <u>BoldReports.Mvc4</u> <u>BoldReports.Mvc5</u> | The BoldReports.Mvc4, and BoldReports.Mvc5 packages contains HTML Helpers for Report Viewer and Report Designer components to build server-side dynamic websites based on Microsoft's ASP.NET MVC. |
| Server Side Helper | Syncfusion.ReportDesigner.AspNet.Mvc Syncfusion.ReportViewer.AspNet.Mvc | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

Refer the ASP.NET MVC Embedded Reporting Tools migration steps [here](#).

Migrate ASP.NET Embedded Reporting Tools

The following provides step-by-step instructions for migrating your ASP.NET reporting application.

| Purpose | Essential Studio Packages | Bold Reports Package | Description |
|------------------------|--------------------------------------------------------------------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Client Side Tag Helper | Syncfusion.AspNet | <u>BoldReports.Webforms</u> | The BoldReports.WebForms package contains HTML Helpers for Report Viewer and Report Designer components to build server-side dynamic websites based on Microsoft's ASP.NET Web Forms. |
| Server Side Helper | Syncfusion.ReportDesigner.AspNet Syncfusion.ReportViewer.AspNet | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

Refer the ASP.NET Embedded Reporting Tools migration steps [here](#).

Migrate WPF Embedded Reporting Tools

The following provides step-by-step instructions for migrating your WPF reporting application.

| Platform | Purpose | Essential Studio Packages | Bold Reports Package | Description |
|----------|---------------------|----------------------------------------------------------------------------------------------|------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| WPF | Components Packages | Syncfusion.ReportControls.WPF Syncfusion.ReportViewer.WPF Syncfusion.ReportWriter.Base | <u>BoldReports.Wpf</u> | BoldReports.Wpf is a .NET Report Viewer, Report Writer control package to view and export RDL/RDLC reports within a .NET application. |

Refer the WPF Embedded Reporting Tools migration steps [here](#).

Migrate UWP Embedded Reporting Tools

The following provides step-by-step instructions for migrating your UWP reporting application.

| Purpose | Essential Studio Packages | Bold Reports Package | Description |
|---------------------|-------------------------------|------------------------|--------------------------------------------------------------------------------|
| Components Packages | Syncfusion.SfReportViewer.UWP | <u>BoldReports.UWP</u> | BoldReports.UWP is a Report Viewer control package to display RDL/RDLC reports |

| | | | |
|--------------------|-----------------------------|----------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| | | | within the Universal Windows Platform application. It builds the server-side implementations for Report Viewer component. |
| Server Side Helper | Syncfusion.Web.ReportViewer | <u>BoldReports.Web</u> | BoldReports.Web is a server-side helper package to build Web API service for Report Viewer and Report Designer components. |

Refer the UWP Embedded Reporting Tools migration steps [here](#).

Uninstallation of Reporting Tools

You can uninstall the Embedded Reporting Tools from the control panel through the following steps.

1. On the Start menu, click Control Panel.
2. In the Control Panel, click Programs and Features.
3. Search Bold Reporting Tools in the Name list and click it.
4. Click **Uninstall**.

![Uninstallation of Reporting Tools](/static/assets/embedded-reporting-tools/images/uninstallation/control-panel-uninstall.png)

5. Click **Yes** to confirm the uninstallation of Reporting Tools.
6. The Embedded Reporting Tools setup will be uninstalled and re-direct to the Embedded Reporting Tools [Uninstallation page](#).

Frequently asked questions

This section helps to get the answer for the frequently asked questions in [Embedded Reporting Tools](#). We discussed the below topics in this section,

- [How to install Embedded Reporting Tools in silent mode?](#)
- [Is Bold Reports Embedded Reporting Tools or Viewer SDK requires additional licensing when running application with Azure App service?](#)
- [How to provide the permission for user to access the SSRS Report Server reports?](#)
- [How to use BoldReports ReportViewer with EJ2 controls?](#)
- [How to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x?](#)
- [What are all the SSRS versions supported in Bold Reports?](#)
- [Is Bold Reports have a centralized report authoring system?](#)
- [How to get the user from database and pass the user as parameter for filtering the data from database?](#)

- [Is Bold Reports Embedded Reporting Tools supports with .NET Core and Docker on Linux?](#)
- [How to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.1?](#)
- [Licensing procedure for deployment](#)
- [Is it possible to create RDLC reports from business object datasource?](#)
- [Can we use the Report Designer component from Syncfusion Community license?](#)
- [How to add header authorization for report service?](#)
- [Can the Bold Reports be used with .NET Core Linux and macOS?](#)
- [Is it possible to use the .NET class objects with Report Viewer?](#)
- [Is theme studio available for Bold Reports to generate the custom theme?](#)
- [Is it possible to create WPF .Net Core application with Report Viewer?](#)
- [Why should you migrate to Bold Reports from Syncfusion Report platform](#)
- [Is it possible to load the million records report with Report Viewer and Report Writer?](#)
- [Can specify ReportServer URL in Web API controller?](#)
- [What is the difference between Report Service URL and Report Server URL?](#)
- [How to change the data source for a report using parameter?](#)
- [How to use BoldReports ReportViewer with EJ1 controls?](#)

Troubleshooting Embedded Reporting Tools

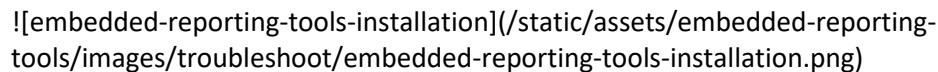
If you are facing issues with launching the **Bold Reports Embedded Reporting Tools** local samples follow the below troubleshooting mechanism.

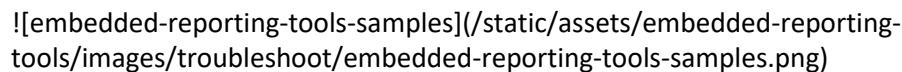
- **If you face problem regarding controlled folder access(Ransomware protection) permission for Bold Reports Embedded Reporting Tools?**

We recommend you to refer how to enable permission to [allow-access-protected-folders](#).

- **Even though you have allowed permission for blocking apps, still not able to launch the Embedded Reporting Tools Sample Browser?**

We recommend you to ensure the below folders present in the installation path **C:\Program Files (x86)\Bold Reports\Embedded Reporting Tools** and **C:\Users\Public\Documents\Bold Reports\Embedded Reporting Tools\Samples**.





- **Even you have properly shipped folders, but still not able to launch Bold Reports Embedded Reporting Tools Sample Browser?**

Open command prompt in administrator mode and navigate to the path **C:\Program Files (x86)\Bold Reports\Embedded Reporting Tools\Utilities\StartSampleBrowserReportingTools** and run anyone of the below command through the **StartSampleBrowserReportingTools.exe**.

| Platform | Arguments |
|----------|-----------|
|----------|-----------|

Controlled folder access and the protected file usage Steps to allow Bold Reports Embedded Reporting Tools to access protected folders

| | |
|--------------|---------------------------------------------------|
| JavaScript | StartSampleBrowserReportingTools.exe "JAVASCRIPT" |
| Angular | StartSampleBrowserReportingTools.exe "ANGULAR" |
| ASP.NET | StartSampleBrowserReportingTools.exe "ASPNET" |
| ASP.NET MVC | StartSampleBrowserReportingTools.exe "ASPNETMVC" |
| ASP.NET Core | StartSampleBrowserReportingTools.exe "ASPNETCORE" |

After running the above command, ensure the System tray contains **IIS EXPRESS.exe** application which hosts the samples.

![IIS EXPRESS.exe](/static/assets/embedded-reporting-tools/images/troubleshoot/IIS EXPRESS.png)

![iis-hosted](/static/assets/embedded-reporting-tools/images/troubleshoot/iis-hosted.png)

- **If none of the above steps doesn't help to launch the Samples?**

You can able to find the error log file in the path **C:\Program Files (x86)\Bold Reports\Embedded Reporting Tools\Utilities\StartSampleBrowserReportingTools**, kindly [contact us](#) by creating a support ticket and share the generated log file with us, we will reach you soon.

Controlled folder access and the protected file usage

[Controlled folder access](#) helps you protect valuable data from malicious apps and threats, such as [ransomware](#). Controlled folder access is included with Windows 10 and Windows Server 2019. When you enable the ransomware protection in your machine, the applications will be restricted to access the protected folders in your machine.

If the Bold Reports Embedded Reporting Tools or its related executable are not allowed to access the protected folders in your machine when the ransomware protection is enabled, follow the below-mentioned steps to provide access to Bold Reports application for accessing the protected folders.

You will receive notifications from Windows when an application is blocked from accessing the protected folders. For example, the following message will be displayed when you are trying to preview the dashboard with ransomware protection enabled.

![app-block-notification](/static/assets/embedded-reporting-tools/images/enable-permission/app-block-notification.png)

Steps to allow Bold Reports Embedded Reporting Tools to access protected folders

- Open the Ransomware protection in Windows security using the start menu.

![search-controlled-folder](/static/assets/embedded-reporting-tools/images/enable-permission/search-controlled-folder.png)

- Click the option **Allow an app through Controlled folder access**.

![allow-protection](/static/assets/embedded-reporting-tools/images/enable-permission/allow-protection.png)

- The list of applications will be shown that can access the protected folders in your machine and click the Add an allowed app option.

![add-blocked-app](/static/assets/embedded-reporting-tools/images/enable-permission/add-blocked-app.png)

- It will help you choose which application should be allowed to access the protected folder by showing recently blocked apps and all apps.

![recently-blocked](/static/assets/embedded-reporting-tools/images/enable-permission/recently-blocked.png)

You can click any one of the options and choose the Bold Reports Embedded Reporting Tools related applications.

![recently-blocked-apps](/static/assets/embedded-reporting-tools/images/enable-permission/recently-blocked-apps.png)

- Allow the iisexpress and the Bold Reports Embedded Reporting Tools related apps to access your protected folders if the application is blocked in your machine.

| Application | Path |
|---------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| iisexpress.exe | C:\Program Files (x86)\IIS Express |
| Bold Reports Embedded Reporting Tools Sample Browser Launcher | C:\Program Files (x86)\Bold Reports\Embedded Reporting Tools\Utilities\StartSampleBrowserReportingTools\StartSampleBrowserReportingTools.exe |

- Now, you can continue to use the Bold Reports Embedded Reporting Tools without blocking any operations to your protected folders.

References

- <https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/enable-controlled-folders>
- <https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/customize-controlled-folders#allow-specific-apps-to-make-changes-to-controlled-folders>

Silent installation

The silent installation steps are applicable only for version 1.2.7 and below.

1. Double click the Embedded Reporting Tools setup.
2. Embedded Reporting Tools setup will be extracted in temp location (%temp%).

![Silent Setup Temp Location](/static/assets/embedded-reporting-tools/images/installation/silent-installation-setup-path.png)

3. Copy that extracted Embedded Reporting Tools setup to some other location and cancel the installation.
4. Open the command prompt with administrative privileges and run the extracted Embedded Reporting Tools setup with the following arguments.

Arguments:

```
/Install silent /InstallPath:{InstallationPath} /pidkey:{unlock_key} /isdesktopshortcut:{TRUE or FALSE}/Log "{LogFile Path}\filename.log"
```

Example:

```
/Install silent /InstallPath:C:\Program Files (x86)\New\Report /pidkey:@1243453sdffdfvv /isdesktopshortcut:TRUE /Log "C:\Program Files (x86)\New\Install.log"
```

![Silent Installation](/static/assets/embedded-reporting-tools/images/installation/silent-installation.png)

5. Now, Embedded Reporting Tools setup has been installed in silent mode.

Is Bold Reports Embedded Reporting Tools or Viewer SDK requires additional licensing when running application with Azure App service

No, if you are already having the Bold Reports for your team, then you do not have a need to buy additional license for using your application with Azure App service. As per our instruction, we should register the license token in startup of application as explained in the following documentation.

[License Token](#)

How to provide the permission for user to access the SSRS Report Server reports

The user what we are using for the **ReportServerCredential** that should have permission to get the report, datasource , resources from server. So, we make sure we are having the content manager permission available for the folder from where are going to get the reports and the user has the permission to access the Report Server.

[Site Setting of Report Server](#)

Within the SSRS website, the first item to setup is to create system level permissions; these permissions are assigned to the main administrators of SSRS and the "power" users who publish reports. Two main roles, System Administrator and System User are predefined. Assignment to these roles is made by

clicking on Site Setting in the upper right corner of the report server site; next click on the Security link from the left menu.

![Site Setting of Report Server](/static/assets/embedded-reporting-tools/images/ssrs-enable/site-setting-image.png)

Permission of user

Clicking on the Edit option allows you to add, edit, or remove the roles assigned to the user or group as displayed in the below figure. The System Administrator role is reserved for those who need to have full control over the Report Server whereas the System User role is applied to users / groups who are power users of the Report Server.

![User Permission](/static/assets/embedded-reporting-tools/images/ssrs-enable/user-permission.png)

Folder

Click on the Manage Folder button upper right corner of the Report Server.

![Folder](/static/assets/embedded-reporting-tools/images/ssrs-enable/folder-permission.png)

Permission for Folder

Provide the permission for the Group or users

![Folder Permission](/static/assets/embedded-reporting-tools/images/ssrs-enable/folder-permission2.png)

Permission of user with Folder

Provide the Role for the user

![Folder Permission for user](/static/assets/embedded-reporting-tools/images/ssrs-enable/folder-permission3.png)

How to use the ReportViewer along with EJ2 controls

We can able to use the ReportViewer control in EJ2 controls. We must add or import the compatibility styles references of ReportViewer control and Essential JS 2 components with the order used in this article.

Angular

You can the refer below help documentation for how to create a ReportViewer control in Angular platform.

- [Angular ReportViewer getting started](#)

You can refer the below help documentation for how to create Angular application with EJ2 controls.

- [EJ2 control getting started](#)

Find the reference below of using the order and compatibility,

```
'typescript
```

```
"styles": [
```

```
  "src/styles.css",
```

```
"./node_modules/@syncfusion/ej2-base/styles/material.css",
"./node_modules/@syncfusion/ej2-buttons/styles/material.css",
"./node_modules/@boldreports/javascript-reporting-
controls/Content/bold.widgets.core.compatibility.min.css",
"./node_modules/@boldreports/javascript-reporting-
controls/Content/material/bold.theme.compatibility.min.css",
"./node_modules/@boldreports/javascript-reporting-
controls/Content/material/bold.reports.all.min.css"
]
`
```

JavaScript, ASP.NET Webforms, ASP.NET MVC, ASP.NET Core

You can find the below help documentation for how to create a ReportViewer control in various platform.

- [JavaScript Reportviewer getting started](#)
- [ASP.NET Webforms ReportViewer getting started](#)
- [ASP.NET MVC ReportViewer getting started](#)
- [ASP.NET Core ReportViewer getting started](#)

You can find the below help documentation for how to create various platform with EJ2 controls.

- [JavaScript EJ2 getting started](#)
- [ASP.NET Webforms EJ2 getting started](#)
- [ASP.NET MVC EJ2 getting started](#)
- [ASP.NET Core EJ2 getting started](#)

Find the reference below of using the order and compatibility.

```
`html
<link href="https://cdn.boldreports.com/2.2.32/content/bold.widgets.core.compatibility.min.css"
rel="stylesheet" />

<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.theme.compatibility.min.css"
rel="stylesheet" />
`
```

We need to refer the EJ2 script, theme before referring the ReportViewer script and themes. Also, we have to add the below highlighted compatibility style scripts to avoid the style overlapping problem with ReportViewer control and EJ2 controls

```
`html
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```
<title>BoldReports ReportViewer and EJ2 controls</title>
@ Syncfusion Essential JS 2 Styles @
<link rel="stylesheet" href="https://cdn.syncfusion.com/ej2/styles/compatibility/material.css" />
@ BoldReports Styles @
<link href="https://cdn.boldreports.com/2.2.32/content/bold.widgets.core.compatibility.min.css"
rel="stylesheet" />
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.theme.compatibility.min.css"
rel="stylesheet" />
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
@Default scripts@
<script src="http://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="http://cdnjs.cloudflare.com/ajax/libs/jquery-easing/1.3/jquery.easing.min.js"
type="text/javascript"></script>
<script src="http://cdn.syncfusion.com/js/assets/external/jsrender.min.js"></script>
@ Syncfusion Essential JS 2 Scripts @
<script src="https://cdn.syncfusion.com/ej2/dist/ej2.min.js"></script>
@ BoldReports Scripts @
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<!-Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<!-Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
<!-Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
</head>
`
```

[Adding script compatibility for ASP.NET Core](#)

Add compatibility, use the following code in the **Layout.cshtml** page. Since BoldReports components and EJ2 controls have same library names to perform different actions, conflicts may occur when you refer these both controls in same application. To overcome this, extend these libraries in ej namespace in ASP.NET Core platform.

```
'html
<script>
var dataCopy = Object.assign({}, ej.data);
$.extend(ej, Syncfusion);
$.extend(ej.data, dataCopy);
</script>
`
```

[How to use the Bold Reports with ASP.NET Core 3.x](#)

We are using **Json.NET serializer** for Report Service which has been removed from ASP.NET Core 3.0 shared framework. So, you have to use **AddNewtonsoftJson()** with services to works with **Json.NET serializer** as per the migration information from .NET Core 3.0

[Migrate from ASP.NET Core 2.2 to 3.0](#)

- Add a package reference to **Microsoft.AspNetCore.Mvc.NewtonsoftJson**
- Update **Startup.ConfigureServices** to call **AddNewtonsoftJson**.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    ...
    ...
    services.AddNewtonsoftJson();
}
```

[What are all the SSRS versions are supported in Bold Reports](#)

Find the list of SSRS server versions are supported in Bold Reports.

| Platform | Supported SSRS version |
|----------|------------------------|
|----------|------------------------|

Centralized report authoring

Is Bold Reports have a centralized report authoring system

| | |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ASP.NET Core | SQL Server Reporting Services 2017 SQL Server Reporting Services 2019 |
| ASP.NET MVC | SQL Server Reporting Services 2008 SQL Server Reporting Services 2008 R2 SQL Server Reporting Services 2012 SQL Server Reporting Services 2014 SQL Server Reporting Services 2016 SQL Server Reporting Services 2017 SQL Server Reporting Services 2019 |
| WPF | SQL Server Reporting Services 2008 SQL Server Reporting Services 2008 R2 SQL Server Reporting Services 2012 SQL Server Reporting Services 2014 SQL Server Reporting Services 2016 SQL Server Reporting Services 2017 SQL Server Reporting Services 2019 |

You will use the ASP.NET Core or ASP.NET MVC services for other platforms such as JavaScript, Angular, React, and UWP platforms. So, do not have separate support list for the other platforms.

Centralized report authoring

Centralized authoring explains how to use the report with authors and multiple developers with several access.

Is Bold Reports have a centralized report authoring system

Yes, Bold Reports having On-Premise and Cloud version Report Server to have the reports in centralized report authoring.

We can refer more details of this from [Bold Report Cloud](#) and [Bold Reports On-Premise](#).

Get the user from database and pass the user as parameter for filtering the data from database

Find the following steps to get the user from database and pass the user as parameter for filtering the data from database.

1. Create a dataset with getting the user id from database as shown in the following image.

![User id dataset](/static/assets/javascript/report-viewer/images/how-to/user-id-parameter/user-id-dataset.png)

2. Pass the user id field value from dataset to parameter available value as shown in the following image.

![User id parameter](/static/assets/javascript/report-viewer/images/how-to/user-id-parameter/user-id-parameter.png)

3. Create a new dataset with filtering the data based on the parameter as shown in the following image.

![User id filtering](/static/assets/javascript/report-viewer/images/how-to/user-id-parameter/user-id-filter.png)

Does Bold Reports Embedded Reporting Tools support .NET Core and Docker on Linux
System.Drawing native dependencies

install

Does Bold Reports Embedded Reporting Tools support .NET Core and Docker on Linux

Yes, Bold Reports Embedded Reporting Tools support .NET Core on Linux Docker. You have to ensure the following details while using our component with Docker environment.

We are using the System.Drawing to measure text size for CanGrow feature supports Textbox ReportItem and it requires native libgdiplus library, which does not contain in default microsoft/dotnet image. So, you must add native libgdiplus library with install command as follows in DockerFile.

`command

```
install System.Drawing native dependencies
RUN apt-get update \
&& apt-get install -y --allow-unauthenticated \
libc6-dev \
libgdiplus \
libx11-dev \
&& rm -rf /var/lib/apt/lists/*
`
```

The sample docker file can be downloaded from [here](#).

How to use the Bold Reports with ASP.NET Core 3.1

We are using Json.NET serializer for Report Service which has been removed from ASP.NET Core 3.1 shared framework. So, you have to use AddNewtonsoftJson() with services to works with Json.NET serializer as per the migration information from .NET Core 3.1

[Migrate from ASP.NET Core 2.2 to 3.1](#)

- Add a package reference to Microsoft.AspNetCore.Mvc.NewtonsoftJson
- Update Startup.ConfigureServices to call AddNewtonsoftJson.

`csharp

```
public void ConfigureServices(IServiceCollection services)
{
...
...
services.AddNewtonsoftJson();
}
`
```

Does Bold Reports have any licensing procedure for deployment
dependencies

install System.Drawing native

Does Bold Reports have any licensing procedure for deployment

Yes, you need licensing token to be registered for Bold Reports development environment. If you are using the Bold reports component in the application level, even if you use the dependencies of the installed build, the licensing will not be imported. The following licensing error will be displayed, if the license token is missing in your projects.

This application was built using a trial version of Bold Reports. Include a valid license to remove this license validation message permanently. You can also obtain a free 30 days evaluation license to temporarily remove this message during the evaluation periods.

You need to register the license tokens in your projects. To learn about registering the license token in your projects, refer to this [how to register the Bold Reports license token](#) section.

Is it possible to create the RDLC reports using the business object data source in Report Designer

The Bold Report Designer does not have support to create the RDLC reports using the business object data source. In Bold Report Designer, RDLC reports can be created and the data can be assigned using the JSON array collection only. If you need to create RDLC report using the business object data source, then refer to this [RDLC report creation](#) section.

Can you use the Report Designer component from Syncfusion community license

No, Syncfusion community license cannot be used for Report Designer and you can use your community license for Report Viewer SDK access only. You should have Bold Reports Embedded plan to use Report Designer in application.

![Embedded Reporting Tools download page](/static/assets/embedded-reporting-tools/images/faq/embedded-reporting-tools.png)

How to add a header authorization for report service

You can add the headers from client side to server side using the **ajaxBeforeLoad** event in our Report Viewer component. You can find the following help documentation for how to create a ReportViewer control in various platform.

- [Angular ReportViewer](#)
- [JavaScript ReportViewer](#)
- [ASP.NET Webforms ReportViewer](#)
- [ASP.NET MVC ReportViewer](#)
- [ASP.NET Core ReportViewer](#)

Can the Bold Reports be used with ASP.NET Core on Linux and macOS

Yes, you can make use of our Bold Reports Embedded Reporting tools with Linux and macOS. You have to ensure the following while using our component.

You are using the **System.Drawing** to measure text size for **CanGrow** feature supports available Textbox ReportItem and it requires native libgdiplus library. So, you should install the libgdiplus dependent library, which is not available with non-Windows operating systems.

Linux

Install the libgdiplus by executing the following commands at a terminal (command) prompt:

```
'bash  
sudo apt install libc6-dev  
sudo apt install libgdiplus  
'
```

macOS

Use the Homebrew ("brew") package manager for installing libgdiplus. After installing brew, install the libgdiplus by executing the following commands at a terminal (command) prompt:

```
'bash  
brew update  
brew install mono-libgdiplus  
'
```

See Also

[.NET Core dependencies requirements on Linux for System.Drawing.Common assembly](#)

[.NET Core dependencies requirements on macOS for System.Drawing.Common assembly](#)

[System.Drawing for .NET Core from NuGet packages](#)

Is it possible to use the .NET class objects in ReportViewer

Yes, it is possible to use the .Net class objects as the DataSource in BoldReports ReportViewer. For this, you have to use the `ProcessingMode` property with `ProcessingMode.Local` and your objects need to pass using the `DataSources` property along with the data set name used with RDLC or RDL report.

![RDLC dataset editor](/static/assets/javascript/report-viewer/images/faq/rdlc-dataset-editor/dataset-editor.png)

See Also

[.NET Class objects in Angular ReportViewer](#)

[.NET Class objects in JavaScript ReportViewer](#)

[.NET Class objects in React ReportViewer](#)

[.NET Class objects in ASP.NET Core ReportViewer](#)

[.NET Class objects in ASP.NET MVC ReportViewer](#)

[.NET Class objects in ASP.NET Web Forms ReportViewer](#)

[.NET Class objects in WPF ReportViewer](#)

[.NET Class objects in UWP ReportViewer](#)

Is theme studio available for Bold Reports to generate the custom theme

No, theme studio is not available for Bold Reports to generate the custom theme. As of now you have to contact our Support team with your requirement to get customized theme.

Is it possible to create a WPF .Net Core application with Bold Report Viewer

Yes, it is possible to create a WPF .Net Core application with Bold Report Viewer. For creating a WPF .Net Core application with Bold Report Viewer, you need to ensure that Visual Studio 2019 is installed in your machine.

Please follow these mentioned steps to create a WPF .Net Core application with Bold Report Viewer:

1. Open the Visual Studio 2019 and select **Create a new project**.
 2. Go to **Installed > Visual C# > Windows Desktop**.
 3. Select **WPF App (.NET Core)**, then click **NEXT**.
 4. Change the application name, and then click **Create**.
- ![WPF NetCore application project template](/static/assets/embedded-reporting-tools/images/faq/wpf-net-core-report-viewer.png)
5. Visual Studio creates the project and opens the designer for the default application window named as **MainWindow.xaml**.

You can follow the remaining steps from the [Getting started](#) section to complete the process of creating WPF .Net Core application with Bold Report Viewer.

Why should you migrate to Bold Reports from Syncfusion Report Platform

As per our plan to have a separate suite for the reporting components, we have introduced Bold Reports from Syncfusion. Due to the migration to Bold reports, there will be no further feature updates for Syncfusion Report Platform. So, you need to consider using our Bold Reports for your developments.

Is it possible to load the million records report with Report Viewer and Report Writer

Yes, you can load the million records reports with Bold Reports Report Viewer and Report Writer components.

You should use **EnableVirtualEvaluation** and **DisablePageSplitting** API with Report Viewer and Report Writer for handling million records. Find the following reference document.

[Handle larger amount of data with Report Viewer](#)

Can specify ReportServer URL in Web API controller

Yes, you can specify the [reportServerUrl](#) property in Web API controller as shown in the following code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ReportServerUrl = "https://testing.SSRS.Server.com";
}
```

Difference between Report Service URL and Report Server URL

Report Service URL

Bold Reports Reporting components are built with the architecture of client-server technology. You should have server side Web API service with ASP.NET Core or ASP.NET MVC for using our components and need to assign created API URL with [reportServiceURL](#) API for server integration.

You can make use of the following references, to create or use existing Web API service for our Reporting components:

- [Bold Reports Report Server built-in service for Report Viewer](#)
- [Bold Reports Report Server built-in service for Report Designer](#)
- [Create ASP.NET Core Web API for Report Viewer](#)
- [Create ASP.NET Core Web API for Report Designer](#)
- [Create ASP.NET MVC Web API for Report Viewer](#)
- [Create ASP.NET MVC Web API for Report Designer](#)

Report Server URL

Bold Reports Reporting components provides support to use the reports directly from SQL Server Reporting Services (SSRS) and Bold Reports Report Server. If you are going to use the feature of using the reports from SQL Server Reporting Services (SSRS) and Bold Reports Report Server, then you have to provide the server URL information to our Reporting components using the [reportServerURL](#) API for report processing.

You can refer to the following sections to make use of the report server URL, in order to use the reports from Report Server in our Reporting components:

- [Bold Reports Report Server Reports with Report Viewer](#)
- [Bold Reports Report Server Reports with Report Designer](#)
- [SSRS Reports with Report Viewer](#)

How to use the Bold Reports along with EJ1 controls

We are having the Reports component in both EJ1 and Bold Reports products. So, if you referring the common [web.all.min.js](#) then Report Viewer conflict with EJ and Bold Reports product. So you can

specify the component script instead of referring the **web.all.min.js** script in your application when using the EJ1 component along with the Bold Reports component.

For example, if you want to use the EJ1 Grid item along with BoldReports Report Viewer component then you can specify the **ej.grid.min.js**. You can find the following example for your reference.

```
'html
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<!--EJ1 Grid component script reference.-->
<script src="https://cdn.syncfusion.com/18.1.0.42/js/web/ej.grid.min.js"></script>
<!--Bold Reports Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
'
```

Report Viewer SDK

The Report Viewer SDK includes Report Viewer and Writer components for viewing and exporting the paginated SSRS, RDL, and RDLC reports. Embed Report Viewer and Report Writer into your application for easily transforming mass financial data into pixel perfect, paginated reports and distribute reports to vendors, employees, and stakeholders for more accurate business decision-making. These tools are based on open Report Definition Language (RDL) specification used by Microsoft SQL Server Reporting Services.

Key features

[View Reports](#) --- You can preview the already created reports to take a look on generated report within your applications (Javascript, ASP.NET CORE, Angular, ASP.NET, ASP.NET MVC, WPF, and UWP) using Report Viewer.

[Export Reports](#) --- You can export the RDL or RDLC report to popular formats such as PDF, Excel, CSV, PowerPoint, Word, and HTML using the Report Writer library. Also you can export your report from Report Viewer.

See Also

- [System Requirements](#)
- [Setup Installation Steps](#)
- [Demo Samples Details](#)
- [Nuget and Assemblies Details](#)
- [Migration Steps](#)
- [Setup Uninstallation Steps](#)

Overview

Standalone Report Designer is a report designing tool that provides fast, simple and powerful report creation solution. It gives you the ability to design, edit, preview and export reports locally in your machine.

Key features

Separate Application Launchers --- Individual application launchers are provided for RDL and RDLC reports to design and view the respective report types.

Design and Edit Reports --- Users can create and edit the reports, datasets and various types of datasources. You can design unlimited number of reports using a rich selection of report items.

Business user friendly --- The drag-and-drop friendly report designer allows business users to create reports without any help from IT.

Wide variety of data providers --- Standalone Report Designer supports connecting to major data providers such as Microsoft SQL Server, SQL CE, Oracle, XML, OLEDB, ODBC, SASS and Web API for exploring data and designing reports with a wide range of data sources.

Display Reports --- Newly created or existing reports can be previewed using the built-in report preview.

Export Reports --- You can export the RDL / RDLC report to popular formats like PDF, Excel, CSV, PowerPoint, Word HTML and XML.

Interactive Reports --- End users can able to create highly customized report. You can alter the report layout in preview using filters, highlighting and sorting the data.

Easy Setup --- The setup and installation process of the Standalone Report Designer application is quick and easy. You can start designing reports instantly without requiring any configurations.

Overview

Standalone Report Designer is a report designing tool that is simple and easy to use. It features an intuitive UI to design, edit, preview, and export reports locally in your machine.

Key features

Separate Application Launchers: Individual application launchers are provided for RDL and RDLC reports to design and view the respective report types.

Design and Edit Reports: You can create a professional looking report using a rich set of data region, visualization, and basic report items with reporting features such as parameters, expressions, sorting, grouping, filtering, and more.

Business user-friendly: The drag-and-drop friendly report designer allows business users to create reports without any help from IT.

Wide variety of data providers: Standalone Report Designer supports connecting to major data providers such as Microsoft SQL Server, SQL CE, Oracle, XML, OLEDB, ODBC, and Web API for exploring data and designing reports with a wide range of data sources.

Display Reports: Newly created or existing reports can be previewed using the built-in report viewer.

Export Reports: You can export the RDL and RDLC report to popular formats like PDF, Excel, CSV, PowerPoint, Word, HTML, and XML.

Interactive Reports: End users can create the highly customized report. You can alter the report layout in preview using the filtering, highlighting, and sorting the data.

Easy Setup: The setup and installation process of the Standalone Report Designer application is quick and easy. You can start designing reports instantly without requiring any configurations.

System requirements

This topic describes the software and hardware requirements to run the Standalone Report Designer.

Hardware requirements

The following hardware requirements are necessary to run the Standalone Report Designer:

- 2.4 GHZ or faster, 32 bit or 64 bit processor.
- 8 GB RAM for 32 bit or 2 GB RAM for 64 bit.
- 400 MB Hard Disk space (Installation files).

Supported operating systems

- Windows 7+
- Windows Server 2008 R2+

Download and installation

This section briefly describes the steps involved in download and installation of the Standalone Report Designer setup.

Register and download Standalone Report Designer

1. Go to Standalone Report Designer [download](#) page.
2. Fill the form input details to register new account or sign in with existing account.

![Account login page](/static/assets/standalone-report-designer/images/installation/registration.png)

3. Once portal is created, you will be redirected to the download page.

![Product download page](/static/assets/standalone-report-designer/images/installation/download-option.png)

4. Click download, then select the file type to download.

![Standalone Report Designer setup download](/static/assets/standalone-report-designer/images/installation/downloan-setup.png)

Installing Standalone Report Designer

1. Once setup is downloaded, run the installer from the saved location by clicking the Run button or by double-clicking the EXE file.

![Standalone Report Designer Installer Extraction](/static/assets/standalone-report-designer/images/installation/setup-extraction.png)

2. Accept the License Agreement of Standalone Report Designer by clicking the **License Terms and Conditions**.

![Bold Reports license agreement](/static/assets/standalone-report-designer/images/installation/bold-reports-license-agreement.png)

3. Read and accept the **Consent to install CefSharp binaries and dependencies** and click **Next**.

CefSharp is a lightweight .NET wrapper around the Chromium Embedded Framework (CEF) used for WPF web browser control implementations.

![CefSharp license agreement](/static/assets/standalone-report-designer/images/installation/cefsharp-license-agreement.png)

4. Choose the location where you would like to install the Standalone Report Designer setup and click **Install**.

![Path Selection Wizard](/static/assets/standalone-report-designer/images/installation/path-selection-page.png)

5. Now, the installation begins. You can cancel the installation anytime by clicking **Cancel**, if you prefer.

![Progress bar Wizard](/static/assets/standalone-report-designer/images/installation/installation-progress.png)

6. On successful installation, the below screen appears. You can launch the application by clicking the **Launch Application**.

![Finish Wizard](/static/assets/standalone-report-designer/images/installation/installation-finish-wizard.png)

7. The installation wizard adds two separate application launchers **Bold Reports Designer** and **Bold Reports RDLC Designer** to your desktop, they are for creating and editing RDL and RDLC reports, respectively.

![Desktop Shortcuts](/static/assets/standalone-report-designer/images/installation/desktop-shortcuts.png)

Designing reports

The Standalone Report Designer features an easy to use environment that enables both the developers and users to design the reports and perform other operations like - edit, preview, save, and export reports seamlessly without any code. Refer the following user guide sections to get started with the Standalone Report Designer.

[Create a report](#)

The user-friendly environment of the report designer makes it easy for you to design the reports. You can highly customize the report and represent your data using a rich selection of report items.

[Create a RDL report](#)

[Create a RDLC report](#)

[Open a report](#)

Standalone Report Designer provides separate launchers to open the RDL and RDLC reports. You can open and edit the existing reports in your local machine.

[Open report](#)

[Save a report](#)

When you create or edit a report, you will use the save command to save your changes. While saving the report, you can specify the location path and also you can edit the report name.

[Save report](#)

[Preview a report](#)

In Standalone Report Designer a newly created report or an existing report can be previewed using the built-in viewer to ensure that the report design is as expected. In the report preview mode, you can export your report to formats like PDF, Excel, CSV, PowerPoint, Word HTML, and XML.

[Preview report](#)

[Uninstallation of Standalone Report Designer](#)

You can uninstall the Standalone Report Designer from your machine by the following these steps:

1. In the Start menu, search for **Control Panel** and select it to open.
2. In the Control Panel, click **Programs and Features**.
3. Search for **Bold Reports Designer** in the Programs and Features search bar and click Enter.
4. Click **Uninstall**.

![Uninstallation of Standalone Report Designer](/static/assets/standalone-report-designer/images/uninstallation/control-panel-uninstall.png)

5. Click **Yes** to confirm the uninstallation of Bold Reports Designer.

![Uninstallation of Standalone Report Designer](/static/assets/standalone-report-designer/images/uninstallation/uninstall-confirmation.png)

6. The Standalone Report Designer setup will be uninstalled from your machine.

[How to queries for Bold Reports Report Designer](#)

This section helps to get the answer for the frequently asked how to queries for **Report Designer**. Discussed the following topics in this section.

- [How to pass the login user as parameter to Stored Procedure?](#)

- [How to add interactive sorting for tablix and matrix header in report?](#)

How to pass the login user as parameter to Stored Procedure

Use the `=User.UserID` expression to pass the login user as parameter for a stored procedure. Find the following steps to pass the login user value to stored procedure parameter for getting the data from database based on user.

1. Create a dataset using the stored procedure, which having a UserID as parameter from `Dataset` dialog box.
2. Select the dataset from `Data` panel and click the highlighted icon to open context menu with list of options. Select `Parameters...` option in the menu to open `Parameters` dialog box as shown in the following image.

![Select Parameter](/static/assets/javascript/report-designer/images/how-to/stored-procedure-userid-parameter/select-parameters.png)

3. Now, click the highlighted square icon of `UserId` parameter to open context menu with two options and select `Expression...` to open `Expression` dialog box as shown in the following image.

![Open Expression Dialog](/static/assets/javascript/report-designer/images/how-to/stored-procedure-userid-parameter/expression.png)

4. Then, select `User` option under `Built-in-Fields` from the highlighted `Options` combo box as shown in the following image.

![Select UserID](/static/assets/javascript/report-designer/images/how-to/stored-procedure-userid-parameter/user-id.png)

5. Select `UserID` and double click it to set `UserID` value as shown in the following image.

![Select Parameter](/static/assets/javascript/report-designer/images/how-to/stored-procedure-userid-parameter/set-user-id.png)

How to add interactive sorting for tablix and matrix header in the report

Find the following steps for adding the interactive sorting for tablix or matrix report item header

1. Open the report in the report designer and right-click the text box in the column header that you want to add an **interactive sort** button, and then click **Text Box Properties**.

![Interactive sort1](/static/assets/standalone-report-designer/images/how-to/interactive-sorting1.png)

2. Click **User sort** as shown in the following image.

![Interactive sort2](/static/assets/standalone-report-designer/images/how-to/interactive-sorting2.png)

3. Provide the expressions in **sort expression** as shown in the following image.

![Interactive sort3](/static/assets/standalone-report-designer/images/how-to/interactive-sorting3.png)

You can repeat this action for multiple textboxes of the tablix or matrix headers.

Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports Standalone Report Designer.

1. [How to do automatic resizing of Textbox based on content?](#)
2. [How to change the data source for a report using parameter?](#)

How to do automatic resizing of Textbox based on their content

Use the **Can Grow** and **Can Shrink** properties with Textbox report items. This will grow and shrink Textbox height vertically based on their content.

![Visibility property](/static/assets/on-premise/images/report-designer/report-items/textbox/miscellaneous-property.png)

How to change data source for a report using the parameter

A report can be previewed with different data sources by changing the data source value using a parameter. For this, you need to have a parameter in the report that specifies the data source name and database name and also you need to set the data source connection string as expression using the parameter value as reference, as shown in the following image.

![Product download page](/static/assets/standalone-report-designer/images/faq/datasource-expression-connection-string.png)

Bold reporting tools for Blazor

Enterprise-class reporting tools for Blazor development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your Blazor applications.

How to best read this user guide

The best way to get started would be to read the **Getting Started** section of the documentation for the control that you would like to start using first. The **Getting Started** guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major

capabilities of RDL reporting and export the report to PDF, Excel, CSV, Word, PowerPoint, and HTML formats. Some of the key features are:

- Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.
- Easily customize each element of the Report Viewer and provide events for report processing customization.
- Supports jQuery, Angular, React, Ember, Aurelia, PHP, and JSP.

Add Web Report Viewer in Blazor application

This section explains the steps required to add web Report Viewer to a Blazor application.

Prerequisites

Before getting started with bold web report viewer, make sure your development environment includes the following requirements.

- [Visual Studio 2019](#) with ASP.NET and Web Development workloads.
- [.NET Core 3.1](#) Framework.

Create a Blazor application

To get started, create a Blazor Server App template application and name it **BlazorReportingTools** using Visual Studio 2019. If this template is not available for you, refer this [link](#) to configure the system environment.

The source code for this Blazor reporting components app is available on [GitHub](#).

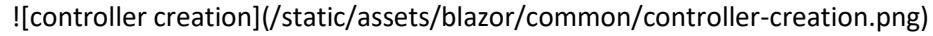
Create a Web API for report processing

In this section, we are going to create a Web API controller that will be used to process the provided RDL reports.

- Install the following [NuGet packages](#) in the created Blazor application. These are used for processing the RDL reports.

| Package | Purpose |
|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BoldReports.Net.Core | Creates Web API service to process the reports. |
| System.Data.SqlClient | Should be referenced in the project when the RDL report renders visual data from the SQL Server or SQL Azure data source based on the RDL design. The package version should be higher than 4.1.0. |
| Microsoft.AspNetCore.Mvc.NewtonsoftJson | ASP.NET Core MVC features that use Newtonsoft.Json . Includes input and output formatter for JSON and JSON Patch. The package version should be higher than 3.1.2. |

- Create an empty API controller by right-clicking the **Data** folder, choosing **Add --> Controller**, and then naming it **BoldReportsAPIController.cs**

![controller creation](//static/assets/blazor/common/controller-creation.png)

- We are going to implement the `IReportController` interface from the namespace `BoldReports.Web.ReportViewer`, which is used to process the reports in the `wwwroot/resources` folder. Refer to the following code sample for RDL report processing.

```
'csharp
using BoldReports.Web.ReportViewer;
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.Caching.Memory;
namespace BlazorReportingTools.Data
{
    [Route("api/{controller}/{action}/{id?}")]
    [ApiController]
    public class BoldReportsAPIController : ControllerBase, IReportController
    {
        // Report viewer requires a memory cache to store the information of consecutive client requests and
        // the rendered report viewer in the server.
        private IMemoryCache _cache;
        // IHostingEnvironment used with sample to get the application data from wwwroot.
        private IWebHostEnvironment _hostingEnvironment;
        public BoldReportsAPIController(IMemoryCache memoryCache, IWebHostEnvironment hostingEnvironment)
        {
            _cache = memoryCache;
            _hostingEnvironment = hostingEnvironment;
        }
        //Get action for getting resources from the report
        [ActionName("GetResource")]
        [AcceptVerbs("GET")]
        // Method will be called from Report Viewer client to get the image src for Image report item.
        public object GetResource(ReportResource resource)
        {
            return ReportHelper.GetResource(resource, this, _cache);
        }
    }
}
```

```
// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from the application folder
    // wwwroot\Resources. sales-order-detail.rdl should be in the wwwroot\Resources application folder.
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\resources\" +
        reportOption.ReportModel.ReportPath + ".rdl", System.IO.FileMode.Open, System.IO.FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}

// Method will be called when report is loaded internally to start the layout process with ReportHelper.

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

[HttpPost]
public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, _cache);
}

// Post action to process the report from the server based on json parameters and send the result back
// to the client.

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
{
    return ReportHelper.ProcessReport(jsonArray, this, this._cache);
}
}
```

- To request the report processing unit properly, we changed the router API attribute to include the controller and action names using `[Route("api/{controller}/{action}/{id?}")]`.
- To invoke this Web API with the controller and action, include that information in the `endPoint` routing in the `Startup.cs` file.

```
'csharp
app.UseEndpoints(endpoints =>
{
    endpoints.MapControllers();
    endpoints.MapBlazorHub();
    endpoints.MapFallbackToPage("/_Host");
});
```

- Bold Reports processes the data between client and server using `Newtonsoft.Json`. Since we need to enable the `Newtonsoft.Json` features in our Blazor application, we need to include the following code section in the `Startup.cs` file's `ConfigureServices` method.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddRazorPages();
    services.AddServerSideBlazor();
    services.AddSingleton();
    services.AddControllers().AddNewtonsoftJson();
}
```

If you are facing the following issue, be sure to install the NuGet package `Microsoft.AspNetCore.Mvc.NewtonsoftJson` with version `3.1.2` or above.

“`IMvcBuilder`” does not contain a definition for “`AddNewtonsoftJson`” and no accessible extension method “`AddNewtonsoftJson`” accepting a first argument of type “`IMvcBuilder`” could be found (are you missing a using directive or an assembly reference?).

[Initialize the Report Viewer](#)

In this section, we are going to integrate Bold Reports JavaScript controls by creating an interop file to initialize the report viewer with basic parameters.

- Create a `Data/BoldReportOptions.cs` class with the following code to hold the RDL report rendering properties.

[`Data/BoldReportOptions.cs`]

```
'csharp
namespace BlazorReportingTools.Data
```

```
{
public class BoldReportViewerOptions
{
    public string ReportName { get; set; }
    public string ServiceURL { get; set; }
}
}
```

- Create a `boldreports-interop.js` file inside the `wwwroot/scripts` folder and use the following code snippet to invoke the Bold Report Viewer JavaScript control.

```
`csharp
// Interop file to render the Bold Report Viewer component with properties.
window.BoldReports = {
    RenderViewer: function (elementID, reportViewerOptions) {
        $("#" + elementID).boldReportViewer({
            reportPath: reportViewerOptions.reportName,
            reportServiceUrl: reportViewerOptions.serviceURL
        });
    }
}
```

- Create a folder named `resources` inside the `wwwroot` folder in your application to store the RDL reports. Now, add any RDL reports that have already been created to the newly created folder.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded [here](#). You can add the reports from the Bold Reports installation location. For more information, refer to the [samples and demos](#) section of the Bold Reports documentation.

- Reference the following online CDN links along with the `boldreports-interop.js` interop file in the head section of `Pages/_Host.cshtml` to use our JavaScript reporting controls in the Blazor application.

```
`html
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
      rel="stylesheet" />
```

```

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>

<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>

<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>

<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>

<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>

<!-- Blazor interop file -->
<script src="~/scripts/boldreports-interop.js"></script>

`
```

- Inject [IJSRuntime](#) and invoke this JavaScript interop with the sales-order-detail.rdl report and the created [BoldReportsAPI](#) URL in the [Pages/Index.razor](#) file to visualize the report using our viewer.

[Pages/Index.razor]

```

`csharp

@page "/"
@using Microsoft.JSInterop
@using Microsoft.AspNetCore.Components
@inject IJSRuntime JSRuntime
@using BlazorReportingTools.Data;
```

```
<div id="report-viewer" style="width: 100%;height: 950px"></div>
@code {
// ReportViewer options
BoldReportViewerOptions viewerOptions = new BoldReportViewerOptions();
// Used to render the Bold Report Viewer component in Blazor page.
public async void RenderReportViewer()
{
    viewerOptions.ReportName = "sales-order-detail";
    viewerOptions.ServiceURL = "/api/BoldReportsAPI";
    await JSRuntime.InvokeVoidAsync("BoldReports.RenderViewer", "report-viewer", viewerOptions);
}
// Initial rendering of Bold Report Viewer
protected override void OnAfterRender(bool firstRender)
{
    RenderReportViewer();
}
}
```

Here we created and used `BoldReportViewerOptions` to pass the parameters for report rendering. In the future, if we need any additional parameters, we can just include them in the `BoldReportsOptions.cs` file and use it to render the reports

Run the Application

Click `Run` or `F5` button to launch the application. The report will be rendered and displayed as shown in the following screenshot.

`![Report Viewer Output](/static/assets/blazor/report-viewer/report-viewer-output.png)`

See Also

[Bold Reports Licensing](#)

Bold reporting tools for Blazor

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

Key features

Data Sources --- Supports connection to major data providers such as

Microsoft SQL Server, Oracle, OLEDB and **ODBC** for exploring data and design reports with a wide range of data sources.

User friendly Environment --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

Report items --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

Report Parameter --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

Expression --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

Add Web Report Designer in Blazor application

This section explains the steps required to add web Report Designer to a Blazor application.

Prerequisites

Before getting started with bold web report designer, make sure your development environment includes the following requirements.

- [Visual Studio 2019](#) with ASP.NET and Web Development workloads.
- [.NET Core 3.1 Framework](#).

Create a Blazor application

To get started, create a Blazor Server App template application and name it **BlazorReportingTools** using Visual Studio 2019. If this template is not available for you, refer this [link](#) to configure the system environment.

The source code for this Blazor reporting components app is available on [GitHub](#).

Create a Web API for report designing

In this section, we are going to create a Web API controller that will be used to process the designer file and data actions.

- Install the following [NuGet packages](#) in the created Blazor application. These are used for processing the RDL reports.

| Package | Purpose |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BoldReports.Net.Core | Creates Web API service to process the designer file and data actions. |
| System.Data.SqlClient | Should be referenced in the project when the RDL report renders visual data from the SQL Server or SQL Azure data source based on the RDL design. The package version should be higher than 4.1.0. |

| Package | Purpose |
|-----------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Microsoft.AspNetCore.Mvc.NewtonsoftJson | ASP.NET Core MVC features that use Newtonsoft.Json . Includes input and output formatter for JSON and JSON Patch. The package version should be higher than 3.1.2. |

- Create an empty API controller by right-clicking the **Data** folder, choosing **Add --> Controller**, and then naming it **BoldReportsAPIController.cs**

![controller creation](//static/assets/blazor/common/controller-creation.png)

- We are going to implement the **IReportDesignerController** interface from the namespace **BoldReports.Web.ReportDesigner** which contains the required actions and helper methods declaration to process the designer file and data actions. The **ReportDesignerHelper** and **ReportHelper** class contains methods that help to process **Post** or **Get** request from the control and return the response. Refer to the following code sample.

```
'csharp
using BoldReports.Web.ReportViewer;
using Microsoft.AspNetCore.Hosting;
using BoldReports.Web.ReportDesigner;
using Microsoft.Extensions.Caching.Memory;
namespace BlazorReportingTools.Data
{
    [Route("api/{controller}/{action}/{id?}")]
    [ApiController]
    public class BoldReportsAPIController : ControllerBase, IReportDesignerController
    {
        private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
        private IWebHostEnvironment _hostingEnvironment;
        public BoldReportsAPIController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        IWebHostEnvironment hostingEnvironment)
        {
            _cache = memoryCache;
            _hostingEnvironment = hostingEnvironment;
        }
        private string GetFilePath(string itemName, string key)
```

```
{  
    string targetFolder = this._hostingEnvironment.WebRootPath + "\\\";  
    targetFolder += "Cache";  
    if (!System.IO.Directory.Exists(targetFolder))  
    {  
        System.IO.Directory.CreateDirectory(targetFolder);  
    }  
    if (!System.IO.Directory.Exists(targetFolder + "\\\" + key))  
    {  
        System.IO.Directory.CreateDirectory(targetFolder + "\\\" + key);  
    }  
    return targetFolder + "\\\" + key + "\\\" + itemName;  
}  
  
public object GetImage(string key, string image)  
{  
    return ReportDesignerHelper.GetImage(key, image, this);  
}  
  
public object GetResource(ReportResource resource)  
{  
    return ReportHelper.GetResource(resource, this, _cache);  
}  
  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    //You can update report options here  
}  
  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    //You can update report options here  
}  
  
[HttpPost]  
public object PostDesignerAction([FromBody] Dictionary<string, object> jsonResult)  
{  
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null, this._cache);  
}
```

```
}

public object PostFormDesignerAction()
{
    return ReportDesignerHelper.ProcessDesigner(null, this, null, this._cache);
}

public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, this._cache);
}

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

bool IReportDesignerController.SetData(string key, string itemId, ItemInfo itemData, out string
errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        System.IO.File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = System.IO.Path.GetFileName(itemData.PostedFile.FileName);
        }
        using (System.IO.MemoryStream stream = new System.IO.MemoryStream())
        {
            itemData.PostedFile.OpenReadStream().CopyTo(stream);
            byte[] bytes = stream.ToArray();
        }
    }
}
```

```

var writePath = this.GetFilePath(fileName, key);
System.IO.File.WriteAllBytes(writePath, bytes);
stream.Close();
stream.Dispose();
}
}
return true;
}

public ResourceInfo GetData(string key, string itemId)
{
var resource = new ResourceInfo();
resource.Data = System.IO.File.ReadAllBytes(this.GetFilePath(itemId, key));
return resource;
}
[HttpPost]
public void UploadReportAction()
{
ReportDesignerHelper.ProcessDesigner(null, this, this.Request.Form.Files[0], this._cache);
}
}
}
`
```

- To request the report processing unit properly, we changed the router API attribute to include the controller and action names using `[Route("api/{controller}/{action}/{id?}")]`.
- To invoke this Web API with the controller and action, include that information in the `endPoint` routing in the `Startup.cs` file.

```

`csharp
app.UseEndpoints(endpoints =>
{
endpoints.MapControllers();
endpoints.MapBlazorHub();
endpoints.MapFallbackToPage("/_Host");
});
```

- Bold Reports processes the data between client and server using `Newtonsoft.Json`. Since we need to enable the `Newtonsoft.Json` features in our Blazor application, we need to include the following code section in the `Startup.cs` file's `ConfigureServices` method.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddRazorPages();
    services.AddServerSideBlazor();
    services.AddSingleton();
    services.AddControllers().AddNewtonsoftJson();
}
```

If you are facing the following issue, be sure to install the NuGet package `Microsoft.AspNetCore.Mvc.NewtonsoftJson` with version `3.1.2` or above.

“`IMvcBuilder`” does not contain a definition for “`AddNewtonsoftJson`” and no accessible extension method “`AddNewtonsoftJson`” accepting a first argument of type “`IMvcBuilder`” could be found (are you missing a using directive or an assembly reference?).

Initialize the Report Designer

In this section, we are going to integrate Bold Reports JavaScript controls by creating an interop file to initialize the report designer with basic parameters.

- Create a `Data/BoldReportOptions.cs` class with the following code to hold the report designer properties.

[Data/BoldReportOptions.cs]

```
'csharp
namespace BlazorReportingTools.Data
{
    public class BoldReportDesignerOptions
    {
        public string ServiceURL { get; set; }
    }
}
```

- Create a **boldreports-interop.js** file inside the **wwwroot/scripts** folder and use the following code snippet to invoke the Bold Report Designer JavaScript control.

```
'csharp
// Interop file to render the Bold Report Designer component with properties.
window.BoldReports = {
    RenderDesigner: function (elementID, reportDesignerOptions) {
        $("#" + elementID).boldReportDesigner({
            serviceUrl: reportDesignerOptions.serviceURL
        });
    }
}
`
```

- Reference the following online CDN links along with the **boldreports-interop.js** interop file in the head section of **Pages/_Host.cshtml** to use our JavaScript reporting controls in the Blazor application.

```
'html
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
      rel="stylesheet" />
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css"
      rel="stylesheet" />
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
       type="text/javascript"></script>
<script
      src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
      src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-
widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js"></script>
<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>
<!-- Report Designer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js"></script>
<!-- Blazor interop file -->
<script src="~/scripts/boldreports-interop.js"></script>
`
```

- Inject [IJSRuntime](#) and invoke this JavaScript interop with the created **BoldReportsAPI** URL in the **Pages/Index.razor** file to design the report using our designer.

[Pages/Index.razor]

```
'csharp
@page "/"
@using Microsoft.JSInterop
@using Microsoft.AspNetCore.Components
@inject IJSRuntime JSRuntime
@using BlazorReportingTools.Data;
<div id="report-designer" style="width: 100%;height: 950px"></div>
@code {
// ReportDesigner options
BoldReportDesignerOptions designerOptions = new BoldReportDesignerOptions();
// Used to render the Bold Report Designer component in Blazor page.
public async void RenderReportDesigner()
{
designerOptions.ServiceURL = "/api/BoldReportsAPI";
await JSRuntime.InvokeVoidAsync("BoldReports.RenderDesigner", "report-designer", designerOptions);
}
// Initial rendering of Bold Report Designer
protected override void OnAfterRender(bool firstRender)
```

```
{  
    RenderReportDesigner();  
}  
}  
`
```

Here we created and used `BoldReportDesignerOptions` to pass the parameters for report designer. In the future, if we need any additional parameters, we can just include them in the `BoldReportsOptions.cs` file.

Run the Application

Click Run or F5 button to launch the application.

![Report Designer Output](/static/assets/blazor/report-designer/report-designer-output.png)

See Also

[Bold Reports Licensing](#)

Syncfusion reporting tools for Angular

Enterprise-class reporting tools for Angular development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your Angular applications.

How to best read this user guide

The best way to get started would be to read the `Getting Started` section of the documentation for the control that you would like to start using first. The `Getting Started` guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Reporting tools for Angular

Enterprise-class reporting tools for Angular development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your Angular applications.

How to best read this user guide

The best way to get started would be to read the `Getting Started` section of the documentation for the control that you would like to start using first. The `Getting Started` guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the

online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

System Requirements

This topic describes the software and hardware requirements for setting up the development environment of Bold Reports Angular.

Supported Operating Systems

- Windows 7+, 8+
- Windows Server 2008 R2+

Software Requirements

The following software requirements are necessary for setting up the development environment of Bold Reports Angular:

- Microsoft Visual Studio Code
- Internet Information Services (IIS) 7.0+
- [Node JS](#) (version 8.x or 10.x)
- [NPM](#) (v3.x.x or higher)
- Angular 4+

Browser Compatibility

- IE 9+
- Microsoft Edge
- Mozilla Firefox 22+
- Chrome 17+
- Opera 12+
- Safari 5+

See Also

- [Licensing procedure for deployment](#)

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Excel, CSV, Word, PowerPoint, and HTML formats. Some of the key features are:

- Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.

- Easily customize each element of the Report Viewer and provide events for report processing customization.
- Supports jQuery, Angular, React, Ember, Aurelia, PHP, and JSP.

Display SSRS RDL report in Bold Reports Angular Report Viewer

This section explains you the steps required to create your first Angular reporting application in Angular CLI to display already created SSRS RDL report in Bold Reports Angular Report Viewer without using a Report Server, refer to the following steps.

If you are using lower version of [Bold Reports Angular](#) (< v2.2.28), then refer [Getting Started for earlier version](#).

Prerequisites

Before you begin, make sure your development environment includes the following:

- [Node JS](#) (version 8.x or 10.x)
- [NPM](#) (v3.x.x or higher)

Install the Angular CLI

Angular provides the easiest way to set Angular CLI projects using the [Angular CLI](#) tool. To install the CLI application globally to your machine, run the following command in the Command Prompt.

```
'typescript
npm install -g @angular/cli@latest
'
```

To learn more about angular-cli commands, click [here](#).

Create a new application

To create a new Angular application, run the following command in the Command Prompt.

```
'typescript
ng new project-name
E.g : ng new reportviewerapp
'
```

The `ng new` command prompts you for information about features to include in the initial app project. Accept the defaults by pressing the Enter or Return key.

![Accept the initial app project defaults by pressing the Enter or Return key](/static/assets/angular/report-viewer/images/getting-started/angular-initial-app-features.png)

Configure Bold Report Viewer in Angular CLI

Bold reporting tools packages are distributed in NPM package as [@boldreports/angular-reporting-components](#).

1. To configure the Report Viewer component, change the directory to your application's root folder.

Display SSRS RDL report in Bold Reports Angular Report Viewer Configure Bold Report Viewer in Angular CLI

```
'typescript
cd project-name
E.g : cd reportviewerapp
'
```

2. Run the following commands to install the [Bold Reports Angular](#) library.

```
'typescript
npm install @boldreports/angular-reporting-components --save-dev
'
```

3. Also, Install [Bold Reports typings](#) by executing the below command.

```
'typescript
npm install --save-dev @boldreports/types
'
```

4. Register the `@bold-reports/types` under the `typeRoots` and add the typings `jquery` and `reports.all` to the `tsconfig.app.json` file.

```
'js
{
...
...
"compilerOptions": {
...
...
"typeRoots": [
"node_modules/@types",
"node_modules/@boldreports/types"
],
"types": [
"jquery",
"reports.all"
]
},
...
'
```

```
...  
}  
`
```

5. Report Viewer requires `window.jQuery` object to render the component. Import jQuery in the `src/polyfills.ts` file as shown in the following code snippet.

```
`js  
import * as jquery from 'jquery';  
window['jQuery'] = jquery;  
window['$'] = jquery;  
`
```

Adding CSS reference

Add Report Viewer component style (`bold.reports.all.min.css`) as given in the `angular.json` file within the `projectname -> styles` section (Eg. `reportviewerapp -> styles`).

If you are using Angular 6 or lower version project, add the changes in the `angular-cli.json` file.

```
`js  
{  
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",  
  "project": {  
    "name": "reportviewerapp"  
  },  
  "reportviewerapp": [  
    {  
      "root": "src",  
      "outDir": "dist",  
      ...  
      ...  
      "styles": [  
        "styles.css",  
        "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css"  
      ],  
      "scripts": [],  
      ...  
    }  
  ]  
}
```

```
...
}  
`
```

In the previous code, the material theme is used. You can modify the theme based on your application, refer the following syntax: ./node_modules/@boldreports/javascript-reporting-controls/Content/[theme-name]/bold.reports.all.min.css

Adding Report Viewer component

To add the Report Viewer component, refer to the following steps:

1. Open the app.module.ts file.
2. You can replace the following code snippet in the app.module.ts file.

```
`typescript  
import { BrowserModule } from '@angular/platform-browser';  
import { NgModule } from '@angular/core';  
import { BoldReportViewerModule } from '@boldreports/angular-reporting-components';  
import { AppComponent } from './app.component';  
// Report viewer  
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';  
// data-visualization  
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';  
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';  
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';  
@NgModule({  
  declarations: [  
    AppComponent  
  ],  
  imports: [  
    BrowserModule,  
    BoldReportViewerModule  
  ],  
  providers: [],  
  bootstrap: [AppComponent]  
})  
export class AppModule {}
```

3. Open the `index.html` file and refer the following scripts in `<tag>`.

```
'html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
'
```

4. Open the `app.component.html` file and initialize the Report Viewer.

5. You can replace the following code snippet in the `app.component.html` file.

```
'javascript
<bold-reportviewer id="reportViewer_Control" style="width: 100%;height: 950px">
</bold-reportviewer>
'
```

6. Open the `app.component.ts` and replace the following code example.

```
'typescript
import { Component } from '@angular/core';
@Component({
selector: 'app-root',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
title = 'reportviewerapp';
public serviceUrl: string;
public reportPath: string;
```

```
constructor() {
  // Initialize the Report Viewer properties here.
}

}

`
```

If you have faced the issue 'ej' is not defined after the above configuration in Angular CLI latest version 7, refer to the following code snippet in your application where you have rendered Syncfusion Components(model file) to resolve the issue.

```
'typescript
/// <reference types="reports.all" />
`
```

Create Web API service

The Report Viewer requires a Web API service to process the report files. You can skip this step and use the online [Web API services](#) to preview the already available reports or you should create any one of the following Web API services:

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

Adding already created report

If you have created a new service, you can add the reports from the Bold Reports installation location. For more information, refer to [samples and demos](#) section.

1. Create a folder Resources in your Web API application to store the RDL reports and add already created reports to it.
2. Add already created reports to the newly created folder.

In this tutorial, the sales-order-detail.rdl report is used, and it can be downloaded in this [link](#).

Refer to the [create RDL report](#) section for creating new reports.

Set report path and Web API service

To set report path and Web API service, open the app.component.ts file and add the codes as shown in the constructor.

```
'typescript
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
```

```

})
export class AppComponent {
title = 'jsreport-sample';
public serviceUrl: string;
public reportPath: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = '~/Resources/docs/sales-order-detail.rdl';
}
}
`
```

In the above code, the `sales-order-detail.rdl` report and `reportServiceUrl` used from online URL.

Open the `app.component.html` to set `reportPath` and `reportServiceUrl` properties of Report Viewer as in the following.

```

`javascript
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [reportPath] =
"reportPath" style="width: 100%;height: 950px">
</bold-reportviewer>
`
```

Serve the application

To serve the application, follow these steps:

1. Navigate to the root of the application and run the application using the following command.

```
`typescript
```

```
ng serve
```

```
`
```

2. Navigate to the appropriate port `http://localhost:4200` in the browser.
3. Click view option to view the demo.

```
{% tab demoPath="angular/report-viewer/getting-started"
files="app.component.ts,app.component.html,app.module.ts" %}
{% endtab %}
```

See Also

[Render Report Server reports](#)

[Create RDLC report](#)

[Render RDLC reports](#)[Preview report in print mode](#)[Set data source credential for shared data sources](#)[Change data source connection string](#)[Production deployment](#)[List of SSRS server versions are supported in Bold Reports](#)

Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports, set the `reportServerUrl`, `reportPath` and `reportServiceUrl` properties as shown in the following code snippet.

`'html'`

```
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] = "Remote" [reportServerUrl] = "serverUrl" [reportPath] = "reportPath">
</bold-reportviewer>
```

`'`

`'typescript'`

```
import { Component } from '@angular/core';
@Component({
  selector: 'ej-app',
  templateUrl: 'src/reportviewer/reportviewer.component.html',
  styleUrls: ['src/reportviewer/reportviewer.component.css']
})
export class ReportViewerComponent {
  public serviceUrl: string;
  public reportPath: string;
  public serverUrl: string;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.serverUrl = 'http://<servername>/reportserver$instanceName';
    this.reportPath = '/SSRSSamples2/Territory Sales new';
  }
}
```

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of /folder name/report name.

Network credentials for SSRS

The network credentials are required to connect with the specified SSRS Report Server using the Report Viewer. Specify the `ReportServerCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server credential
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
}
```

Set data source credential for shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS server. If the report has any data source that uses credentials to connect with the database, then you should specify the `DataSourceCredentials` for each report data source to establish database connection.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-demo",
    "N@c=Y8s*1&dh"));

}
```

Data source credentials should be added to the shared data sources that do not have credentials in the connection strings.

Build and run the application.

Change data source connection string

You can change the connection string of a report data source before it is loaded in the Report Viewer. The `DataSourceCredentials` class provides the option to set and update the modified connection string as in the following code snippet.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-
    demo","N@c)=Y8s*1&dh","Data Source=dataplatformdemodata.syncfusion.com;Initial
    Catalog=AdventureWorks"));
}
```

The previous code shows an option to change the connection string only, but the class provides multiple options to change data source information. To learn more about this, refer to the [DataSourceCredentials](#) class.

Render linked reports

You can render a linked report that points to an existing report, which is published in the SSRS Report Server. You can also set the parameter, data source, credential, and other properties like normal SSRS reports using the Report Viewer.

```
'html
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] =
"Remote" [reportServerUrl] = "serverUrl" [reportPath] = "reportPath">
</bold-reportviewer>
'
```

```
'typescript
import { Component } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: 'src/Report Viewer/Report Viewer.component.html',
styleUrls: ['src/Report Viewer/Report Viewer.component.css']
})
export class Report ViewerComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/Report Viewer';
this.serverUrl = 'http://<servername>/reportserver$instanceName';
this.reportPath = '/SSRSSamples2/Territory Sales_Link';
}
}
```

```
}
```

The `Territory Sales_Link` is a linked report created for the `Territory Sales` report, which is already available in the SSRS Report Server.

Load SharePoint Server reports

To render SharePoint server reports, set the `reportServerUrl`, `reportPath` and `reportServiceUrl` properties as shown in the following code snippet.

```
'html'
```

```
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] =  
"Remote" [reportServerUrl] = "serverUrl" [reportPath] = "reportPath">  
</bold-reportviewer>
```

```
'
```

```
'typescript'
```

```
import { Component } from '@angular/core';  
  
@Component({  
    selector: 'ej-app',  
    templateUrl: 'src/reportviewer/reportviewer.component.html',  
    styleUrls: ['src/reportviewer/reportviewer.component.css']  
})  
  
export class ReportViewerComponent {  
    public serviceUrl: string;  
    public reportPath: string;  
    public serverUrl: string;  
    constructor() {  
        this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';  
        this.serverUrl = 'http://<servername>/reportserver$instanceName';  
        this.reportPath = 'http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl';  
    }  
}
```

In SharePoint integrated mode, the `reportServerUrl` will be same as your site URL. The `reportPath` is relative to the Report Server URL with the file extension.

Forms credential for SharePoint server

The Forms credentials are required to connect with the specified SharePoint integrated SSRS Report Server using the Report Viewer. Specify the `ReportServerFormsCredential` property in the Web API Controller `OnInitReportOptions` method.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential for server
    reportOption.ReportModel.ReportServerFormsCredential = new
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");
}
```

Set data source credential for shared data sources

The shared data source credentials can be added to the `DataSourceCredentials` property to connect with the database.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential and data source credentials
    reportOption.ReportModel.ReportServerFormsCredential = new
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Data source credentials should be added to shared data sources that do not have credentials in the connection strings.

Build and run the application.

Load Bold Report Server reports

This section explains how to render the Bold Report Server reports in your Angular application. The Bold Report Server contains built-in web API service that allows you to display the reports in your application without creating a new web API. You need a Report Viewer Angular application to do the following steps:

If you don't have Report Viewer application then create the app using the [Getting-started](#) steps.

1. Create the `serviceAuthorizationToken` to connect with Bold Report server built-in service.
2. Run the below command to install the `HTTP` dependencies.

Load Bold Report Server reports

Set data source credential for shared data sources

```
'typescript
npm install @angular/http@latest
`
```

3. Open the `app.module.ts` file and import HTTP module.

```
'typescript
...
import { HttpModule } from '@angular/http';
...
imports: [
  BrowserModule,
  HttpModule
],
providers: [],
bootstrap: [AppComponent]
})
...
`
```

4. Open the `app.component.ts` and create the user token using `http.post`. You can replace the following code example in your application.

```
'js
import { Component } from '@angular/core';
import { Http, Response, Headers } from '@angular/http';
const httpOptions = {
  headers: new Headers({
    'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8'
  })
};
class ReportserviceService {
  private http: Http;
  constructor(http: Http) {
    this.http = http;
  }
}
`
```

Load Bold Report Server reports

Set data source credential for shared data sources

```
}
```

```
async getAsyncToken(serverUrl: string, userName: string, password: string): Promise<string>{
    return await this.http.post(serverUrl + '/get-user-key', {userid : userName,password : password
}).toPromise().then(this.extractData);
}
```

```
private extractData(res: Response): string {
    var token = JSON.parse(res.json().Token);
    return token["tokentype"] + " " + token["accesstoken"];
}
```

```
@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']
})
```

```
export class AppComponent {
    public serviceUrl: string;
    public reportPath: string;
    public serverServiceAuthorizationToken: string;
    public serverApiURL: string;
    private tokenService: ReportserviceService
    constructor(private http: Http) {
        this.tokenService = new ReportserviceService(this.http);
        let self = this
        this.serverURL = 'https://on-premise-demo.boldreports.com/reporting/api/site/site1';
        this.serviceUrl ='https://on-premise-demo.boldreports.com/reporting/reportservice/api/Viewer';
        this.tokenService.getAsyncToken(this.serverURL, "guest@boldreports.com", "demo").then(function
        (data) {
            self.serverServiceAuthorizationToken = data;
            self.reportPath = '/Sample Reports/Product Line Sales';
        });
    }
    onAjaxRequest(event) {
        event.headers.push({ Key: 'serverurl', Value: this.serverApiURL });
    }
}
```

```
}
```

```
,
```

5. Open the `app.component.html` and set `reportPath`, `serverServiceAuthorizationToken`, `reportServiceUrl` properties as in the below code example.

```
`js
<bold-reportviewer
  id="reportViewer_Control"
  [reportServiceUrl]="serviceUrl"
  [serviceAuthorizationToken] = "serverServiceAuthorizationToken"
  [processingMode] = "Remote"
  [reportPath] = "reportPath"
  (ajaxBeforeLoad) = "onAjaxRequest($event)"
  [enableParameterBlockScroller] = "true">
</bold-reportviewer>
`
```

6. Navigate to the root of the application and run the application using the following command.

```
`typescript
ng serve
`
```

Render RDLC report

The data binding support allows you view the RDLC reports that exist on the local file system with JSON array and custom business object data collection. The following steps demonstrates how to render an RDLC report with JSON array and custom business object data collection.

Add the RDLC report `Product List.rdlc` from the Bold Reports installation location to your application `App_Data` folder. For more information, refer to [Samples and demos](#).

Bind data source at client side

To bind the data source at client side, follow these steps;

- Set the RDLC report path to the `reportPath` property.
- Assign the `processingMode` property to `ProcessingMode.Local`.
- Bind the JSON array collection to the `dataSources` property as shown in following code.

```
`html
```

```

<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] =
"Local" [reportPath] = "reportPath">
</bold-reportviewer>
`typescript
import { Component } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: 'src/reportviewer/reportviewer.component.html',
styleUrls: ['src/reportviewer/reportviewer.component.css']
})
export class ReportViewerComponent {
public serviceUrl: string;
public reportPath: string;
public reportData: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'AreaCharts.rdlc"';
this.reportData = [{{
value: [
{ SalesPersonID: 281, FullName: 'Ito', Title: 'Sales Representative', SalesTerritory: 'South West', Y2002: 0,
Y2003: 28000, Y2004: 3018725 },
{ SalesPersonID: 282, FullName: 'Saraiva', Title: 'Sales Representative', SalesTerritory: 'Canada', Y2002:
25000, Y2003: 14000, Y2004: 3189356 },
{ SalesPersonID: 283, FullName: 'Cambell', Title: 'Sales Representative', SalesTerritory: 'North West',
Y2002: 12000, Y2003: 13000, Y2004: 1930885 }
],
name: 'AdventureWorksXMLDataSet'
}];}
}
}
`
```

- Build and run the application.

Bind data source in Web API controller

The following steps help you configure the Web API to render the RDLC report with business object data collection.

1. Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
        };
        datas.Add(data);
    }
}
```

```

Category = "Non-Veg",
Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};
datas.Add(data);
return datas;
}
}
`
```

2. Set the value of the `ProcessingMode` property to `ProcessingMode.Local` and `ReportPath` in the RDLC report location.
3. Bind the business object data values collection by adding a new item to the `DataSources` as shown in the following code snippet.

```

`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;
    reportOption.ReportModel.ReportPath =
        System.Web.Hosting.HostingEnvironment.MapPath("~/App_Data/Product List.rdlc");
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
`
```

Here the **Name** is case sensitive and it should be same as in the data source name in the report definition.

The **Value** accepts **IList**, **DataSet**, and **DataTable** inputs.

Load report as stream

To load report as a stream, create a report stream using the **FileStream** class and assign the report stream to the **Stream** property.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string filePath = System.Web.Hosting.HostingEnvironment.MapPath("~/App_Data/Product List.rdlc");
    ;

    // Opens the report from application App_Data folder using FileStream
    FileStream reportStream = new FileStream(filePath, FileMode.Open, FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
```

In the previous code, the **Product List.rdlc** report is loaded from the **App_Data** folder location.

View report click

You can get the user selected parameter details when users clicks the **ViewReport** button in the parameter block. The **viewReportClick** event allows you handle the **ViewReport** button click at client side as shown in the following code.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[locale]= Remote
(viewReportClick) = "viewReportClick($event)">
</bold-reportviewer>
```

'typescript

```
import { Component, ViewChild } from '@angular/core';
```

```
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
}
viewReportClick(event) {
var reportParams = [];
reportParams.push({ name: 'ReportParameter1', labels: ['SO50756'], values: ['SO50756'] });
event.model.parameters = reportParams;
}
}
`
```

The model property in the event argument has the details of current processing report model.

Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

- Add the subreport and main reports to your application App_Data folder. In this tutorial, the already created reports are used. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section.

Download the SideBySideMainReport.rdl and SideBySideSubReport.rdl reports from [here](#). You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#). The reports used from the installed location requires the NorthwindIO_Reports.sdf database to run, so add it to your application.

- Set the `reportPath` and `reportServiceUrl` properties of the Report Viewer as shown in following code snippet.

```
'html  
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] =  
"Remote" [reportServerUrl] = "serverUrl" [reportPath] = "reportPath">  
</bold-reportviewer>  
'  
  
'ts  
import { Component } from '@angular/core';  
@Component({  
selector: 'ej-app',  
templateUrl: 'src/reportviewer/reportviewer.component.html',  
styleUrls: ['src/reportviewer/reportviewer.component.css']  
})  
export class ReportViewerComponent {  
public serviceUrl: string;  
public reportPath: string;  
public serverUrl: string;  
constructor() {  
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';  
this.reportPath = 'SideBySideMainReport.rdl';  
}  
}  
'
```

- Build and run the application.

Change subreport path

To change the subreport file path, set the `ReportPath` property of `SubReportModel` in the `OnInitReportOptions` method.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
if (reportOption.SubReportModel != null)  
{
```

Render subreport

Set subreport parameter

```
reportOption.SubReportModel.ReportPath =  
System.Web.Hosting.HostingEnvironment.MapPath(@"~/AppData/SubReportDetail.rdl");  
}  
}  
`
```

Set subreport parameter

You can change the parameter default values of a subreport in the **OnReportLoaded** method of the Web API Controller as given in the following code snippet.

```
'csharp  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
if (reportOption.SubReportModel != null)  
{  
reportOption.SubReportModel.Parameters = new BoldReports.Web.ReportParameterInfoCollection();  
reportOption.SubReportModel.Parameters.Add(new BoldReports.Web.ReportParameterInfo()  
{  
Name = "SalesPersonID",  
Values = new List<string>() { "2" }  
});  
}  
}  
`
```

Modify subreport data source connection string

You can change the credential and connection information of the data sources used in the subreport using the SubReportModel in the **OnInitReportOptions** method.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
if (reportOption.SubReportModel != null)  
{  
reportOption.SubReportModel.DataSourceCredentials = new  
List<BoldReports.Web.DataSourceCredentials>();  
reportOption.SubReportModel.DataSourceCredentials.Add(new  
BoldReports.Web.DataSourceCredentials("NorthWind", "Data  
Source=dataplatformdemodata.syncfusion.com;Initial Catalog=Northwind;user id=demoreadonly@data-  
platform-demo;password=N@c)=Y8s*1&dh"));  
}
```

```
}
```

```
}
```

```
'
```

Set subreport data source

You can bind local business object data source collection only for RDLC reports. To specify data source of a RDLC subreport, set the `ReportDataSource` property in the `OnReportLoaded` method.

The RDL report has the connection information in report definition itself, so no need to bind the data source.

1. Add the RDLC subreport and main reports to your application `App_Data` folder. You can download it from [here](#).
2. Set the `reportPath` and `reportServiceUrl` properties of the Report Viewer as shown in following code snippet.

```
'html
```

```
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] = "Local" [reportServerUrl] = "serverUrl" [reportPath] = "reportPath">
```

```
</bold-reportviewer>
```

```
'
```

```
'ts
```

```
import { Component } from '@angular/core';
@Component({
  selector: 'ej-app',
  templateUrl: 'src/reportviewer/reportviewer.component.html',
  styleUrls: ['src/reportviewer/reportviewer.component.css']
})
export class ReportViewerComponent {
  public serviceUrl: string;
  public reportPath: string;
  public serverUrl: string;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.reportPath = 'Product List Main.rdlc';
  }
}
```

- Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
            Category = "Non-Veg",
            Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
            ProductImage = ""
        };
        datas.Add(data);
    }
}
```

```
};

datas.Add(data);

data = new ProductList()

{

ProductName = "Chicken Tikka",
OrderId = "323B62",
Price = 64,
Category = "Non-Veg",
Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
ProductImage = ""

};

datas.Add(data);

return datas;

}

}

`
```

- Bind the business object data values collection to subreport by adding a new item to the **DataSources** as shown in the following code snippet.

```
`csharp

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //Assigning the data source for 'Product List.rdlc'
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
        "list", Value = ProductList.GetData() });
    }
}
```

The data source name is case sensitive, and it should be same as in the report definition.

[Load subreport stream](#)

To load subreport as stream, set the **Stream** property in the **OnInitReportOptions** method.

| | |
|-------------------|------------------------------|
| Report parameters | Set parameter at client side |
|-------------------|------------------------------|

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
if (reportOption.SubReportModel != null)
{
// Opens the report from application App_Data folder using FileStream and loads the sub report stream.

FileStream reportStream = new
FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/App_Data/Product List.rdlc"),
 FileMode.Open, FileAccess.Read);

reportOption.SubReportModel.Stream = reportStream;
}

}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
//Assigning the data source for 'Product List.rdlc'

if (reportOption.SubReportModel != null)
{
reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
"list", Value = ProductList.GetData() });

}
}

`
```

Report parameters

Provides property options to pass or set report parameters default values at run time using the **parameters** property. You can set the report parameters while creating the Report Viewer control in a script or in the Web API Controller.

In this tutorial, the **Sales Order Detail.rdl** report is used, and it can be downloaded from [here](#).

Set parameter at client side

The **parameters** property takes the JSON array value input with parameter details.

- Set the default value data to the **values** property and name of the report parameter to the **name** property.

The parameter name is case sensitive, and it should be same as available in the report definition.

The following code example illustrates how to set a report parameter in the script.

```
'html
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] =
"Remote" [reportPath] = "reportPath" [parameters] = "parameters">
</bold-reportviewer>
'

`typescript
import { Component } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: 'src/reportviewer/reportviewer.component.html',
styleUrls: ['src/reportviewer/reportviewer.component.css']
})
export class ReportViewerComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public parameters: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'Sales Order Detail.rdl';
this.parameters = [
{
name: 'SalesOrderNumber',
labels: ['SO50751'],
values: [SO50751],
nullable: false
}];
}
}
```

- Build and run the application.

Set parameters in Web API Controller

To set parameter default value in the Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    List<BoldReports.Web.ReportParameter> userParameters = new
    List<BoldReports.Web.ReportParameter>();
    userParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = "SalesOrderNumber",
        Values = new List<string>() { "SO50756" }
    });
    reportOption.ReportModel.Parameters = userParameters;
}
'
```

Get report parameter

The **ReportHelper** class provides methods that help you to get the report parameters used in the report. The following helper methods are used to get parameter with or without values.

[Methods](#) | [Description](#)

GetParameters | Returns the parameters used in the current report without the processed values.

GetParametersWithValues | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

```
'csharp
public class ReportsApiController : ApiController, IReportController
{
    Dictionary<string, object> jsonArray = null;
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        jsonArray = jsonResult;
        return ReportHelper.ProcessReport(jsonResult, this);
    }
    ....
    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        var reportParameters = ReportHelper.GetParameters(jsonArray, this);
    }
}
```

Report interaction events

Report loaded

```
List<BoldReports.Web.ReportParameter> setParameters = new  
List<BoldReports.Web.ReportParameter>();  
  
if (reportParameters != null)  
{  
  
foreach (var rptParameter in reportParameters)  
{  
  
setParameters.Add(new BoldReports.Web.ReportParameter()  
{  
  
Name = rptParameter.Name,  
  
Values = new List<string>() { "SO50756" }  
});  
}  
  
reportOption.ReportModel.Parameters = setParameters;  
}  
}  
}  
`
```

Report interaction events

You can handle the report interaction events with reports using the following events.

- ReportLoaded
- ReportError
- ShowError
- Drill through
- Hyperlink

Report loaded

The **reportLoaded** event occurs after the report is loaded and it is ready to start the processing. You can handle the event and specify the data source and parameters at client side. The following sample code loads a report by assigning the report data source input in the **reportLoaded** event.

In this tutorial, the **IndicatorReport.rdlc** report is used. You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

'html

```
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [processingMode] =  
"Local" [reportPath] = "reportPath" (reportLoaded) = "reportLoaded($event)">  
</bold-reportviewer>
```

Report interaction events

Report loaded

```
`typescript
import { Component } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: 'src/reportviewer/reportviewer.component.html',
styleUrls: ['src/reportviewer/reportviewer.component.css']
})
export class ReportViewerComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'IndicatorReport.rdlc';
}
reportLoaded(event) {
let desc2013 = [
{
No: 1, Name: "Carlos Slim", NetWorth: 73.0, Age: 73, CitizenShip: "Mexico", Source: "Telmex,America
Movil, Grupo Carso", RankingStatus: 50, ProfitStatus: 75
},
{
No: 2, Name: "Bill Gates", NetWorth: 67.0, Age: 57, CitizenShip: "United States", Source: "Microsoft",
RankingStatus: 50, ProfitStatus: 75
},
{
No: 3, Name: "Amancio Ortega", NetWorth: 57.0, Age: 57, CitizenShip: "Spain", Source: "Inditex Group",
RankingStatus: 75, ProfitStatus: 75
},
{
No: 4, Name: "Warren Buffett", NetWorth: 53.0, Age: 82, CitizenShip: "United States", Source:
"Berkshire Hathaway", RankingStatus: 25, ProfitStatus: 75
},
```

```
{  
No: 5, Name: "Larry Ellison", NetWorth: 43.0, Age: 68, CitizenShip: "United States", Source: "Oracle  
Corporation", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 6, Name: "Charles Koch", NetWorth: 34.0, Age: 77, CitizenShip: "United States", Source: "Koch  
Industries", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 7, Name: "David Koch", NetWorth: 34.0, Age: 72, CitizenShip: "United States", Source: "Koch  
Industries", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 8, Name: "Li Ka-shing", NetWorth: 32.0, Age: 84, CitizenShip: "Hong Kong/ Canada", Source:  
"Cheung Kong Holdings", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 9, Name: "Liliane Bettencourt", NetWorth: 30.0, Age: 90, CitizenShip: "France", Source: "L'Oreal",  
RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 10, Name: "Bernard Arnault", NetWorth: 29.0, Age: 63, CitizenShip: "France", Source: "LVMH Moet  
Hennessy Louis Vuitton", RankingStatus: 25, ProfitStatus: 25  
});  
let desc2012 = [  
{  
No: 1, Name: "Carlos Slim", NetWorth: 69.0, Age: 72, CitizenShip: "Mexico", Source: "Telmex,America  
Movil, Grupo Carso", RankingStatus: 50, ProfitStatus: 25  
},  
{  
No: 2, Name: "Bill Gates", NetWorth: 61.0, Age: 56, CitizenShip: "United States", Source: "Microsoft",  
RankingStatus: 50, ProfitStatus: 75  
},  
{  
No: 3, Name: "Warren Buffett", NetWorth: 44.0, Age: 81, CitizenShip: "United States", Source:  
"Berkshire Hathaway", RankingStatus: 50, ProfitStatus: 25
```

```
},  
{  
No: 4, Name: "Bernard Arnault", NetWorth: 41.0, Age: 63, CitizenShip: "France", Source: "LVMH Moet  
Hennessy Louis Vuitton", RankingStatus: 50, ProfitStatus: 75  
},  
{  
No: 5, Name: "Amancio Ortega", NetWorth: 37.5, Age: 75, CitizenShip: "Spain", Source: "Inditex Group",  
RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 6, Name: "Larry Ellison", NetWorth: 36.0, Age: 67, CitizenShip: "United States", Source: "Oracle  
Corporation", RankingStatus: 25, ProfitStatus: 75  
},  
{  
No: 7, Name: "Eike Batista", NetWorth: 30.0, Age: 55, CitizenShip: "Brazil", Source: "EBX Group",  
RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 8, Name: "Stefan Persson", NetWorth: 26.5, Age: 64, CitizenShip: "Sweden", Source: "H&M",  
RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 9, Name: "Li Ka-shing", NetWorth: 25.0, Age: 83, CitizenShip: "Hong Kong/ Canada", Source:  
"Cheung Kong Holdings", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 10, Name: "Karl Albrecht", NetWorth: 25.4, Age: 92, CitizenShip: "Germany", Source: "Aldi",  
RankingStatus: 75, ProfitStatus: 25  
}];  
let description = [  
{  
Status: 25, Description: "Has not changed from the previous ranking."  
},  
{  
Status: 50, Description: "Has increased from the previous ranking."  
}
```

```

},
{
Status: 75, Description: "Has decreased from the previous ranking."
}];

event.model.dataSources = [
value: ej.DataManager(desc2013).executeLocal(ej.Query()),
name: "DataSet1"
], {
value: ej.DataManager(desc2012).executeLocal(ej.Query()),
name: "DataSet2"
}, {
value: ej.DataManager(description).executeLocal(ej.Query()),
name: "DataSet3"
}];
}
}
`
```

Report error

When an error occurs in the report processing, it raises the `reportError` event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```

`html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
(reportError) = "onReportError($event)">
</bold-reportviewer>
`
```

```

`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
```

```
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'Sales Order Detail.rdl';
}
onReportError(event) {
alert(event.errmsg);
event.cancel = true;
}
}
`
```

To suppress the default error dialog, set the cancel argument to true.

[Show error](#)

The `showError` event is invoked whenever users click a report item that contains an error in processing. It provides detailed information about the cause of the error. You can hide the default dialog and show your customized dialog.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
(showError) = "onShowError($event)">
</bold-reportviewer>
`

`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
```

```
@Component({
  selector: 'ej-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})

export class AppComponent {
  public serviceUrl: string;
  public reportPath: string;
  public serverUrl: string;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.reportPath = 'Sales Order Detail.rdl';
  }
  onShowError(event) {
    alert("Error code : " + event.errorCode + "\n" +
      "Error Detail : " + event.errorDetail + "\n" +
      "Error Message : " + event.errorMessage);
    event.cancel = true;
  }
}
```

Drill through

When a drill through item is selected in a report, it invokes the `drillThrough` event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
(drillThrough) = "onDrillThrough($event)">
</bold-reportviewer>
```

```
`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
  selector: 'ej-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  public serviceUrl: string;
  public reportPath: string;
  public serverUrl: string;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.reportPath = 'SalesPersonDetails.rdl';
  }
  onDrillThrough(event) {
    event.actionInfo.ReportName = "PersonalDetails";
    event.actionInfo.Parameters = [
      {
        name: 'SalesOrderNumber',
        labels: ['SO50751'],
        values: [SO50751],
        nullable: false
      }];
  }
}
```

Hyperlink

The **hyperlink** event occurs when users click a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
```

```
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
(hyperLink) = "onHyperLink($event)">
</bold-reportviewer>
`  

`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'Customer Support Analysis (Random data).rdl';
}
onHyperLink(event) {
event.cancel = true;
//You can modify the URL here
window.open(event.actionInfo.Hyperlink);
}
}
`
```

Handle post actions

Report processing actions are sent in an Ajax request to exchange data with the Web API service. You can handle post actions event to customize the Ajax requests.

- AjaxBeforeLoad

- AjaxSuccess
- AjaxError

AjaxBeforeLoad

This event can be triggered before an Ajax request is sent to the Report Viewer Web API service. It allows you to set additional headers and custom data in the Ajax request. The following code sample demonstrates adding custom authorization header and passing default parameter values to service.

Add custom header in Ajax request

Initialize the `ajaxBeforeLoad` event in the script and add the authorization token to the `headers` property.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
/ajaxBeforeLoad) = "onAjaxRequest($event)">
</bold-reportviewer>
`
```

```
'typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'Sales Order Detail.rdl';
}
}
```

```
onAjaxRequest(event) {  
    event.headers.push({ Key: 'Authorization', Value: 'demo@123' });  
}  
}  
`
```

In this tutorial, the `Sales Order Detail.rdl` report is used, and it can be downloaded from [here](#).

Get the custom header value from the `HttpContext` header collection using the key name specified at client side.

```
`csharp  
string authenticationHeader;  
public object PostReportAction(Dictionary<string, object> jsonResult)  
{  
    if (jsonResult != null)  
    {  
        //Get client side custom ajax header and store in local variable  
        authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];  
        //Perform your custom validation here  
        if (authenticationHeader == "")  
        {  
            return new Exception("Authentication failed!!!");  
        }  
        else  
        {  
            return ReportHelper.ProcessReport(jsonResult, this);  
        }  
    }  
    return null;  
}
```

Perform your own action to validate the header values.

Pass custom data in Ajax request

Use the `data` property to set custom data to the server in the Ajax request. In the following code sample, parameter values are passed to the server side.

```
`html
```

```
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
/ajaxBeforeLoad) = "onAjaxRequest($event)">
</bold-reportviewer>
`  

`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'Sales Order Detail.rdl';
}
onAjaxRequest(event) {
event.headers.push({ Key: 'Authorization', Value: 'demo@123' });
//Passing custom parameter data to server
event.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
}`
```

The custom data values are stored in the `customData` header key, and you can store them in the local property. The following code sample demonstrates storing parameter values and setting those values to the report in the `OnReportLoaded` method.

```
'csharp
public string DefaultParameter = null;
string authenticationHeader;
public object PostReportAction(Dictionary<string, object> jsonResult)
{
if (jsonResult != null)
{
if (jsonResult.ContainsKey("customData"))
{
//Get client side custom data and store in local variable. Here parameter values are sent.
DefaultParameter = jsonResult["customData"].ToString();
}

//Get client side custom ajax header and store in local variable
authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
//Perform your custom validation here
if (authenticationHeader == "")
{
return new Exception("Authentication failed!!!");
}
else
{
return ReportHelper.ProcessReport(jsonResult, this);
}
}
return null;
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (DefaultParameter != null)
{
//Set client side custom header data
reportOption.ReportModel.Parameters =
JsonConvert.DeserializeObject<List<BoldReports.Web.ReportParameter>>(DefaultParameter);
```

```
}
```

```
}
```

```
'
```

AjaxSuccess

To perform custom action or show user defined message, use the `ajaxSuccess` event on the successful Ajax request.

```
`html
```

```
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"  
[processingMode] = "Remote"  
[reportServerUrl] = "serverUrl"  
[reportPath] = "reportPath"  
(ajaxSuccess) = "onAjaxSuccess($event)">  
</bold-reportviewer>
```

```
'
```

```
`typescript
```

```
import { Component, ViewChild } from '@angular/core';  
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';  
@Component({  
  selector: 'ej-app',  
  templateUrl: './app.component.html',  
  styleUrls: ['./app.component.css']  
})  
export class AppComponent {  
  public serviceUrl: string;  
  public reportPath: string;  
  public serverUrl: string;  
  constructor() {  
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';  
    this.reportPath = 'Sales Order Detail.rdl';  
  }  
  onAjaxSuccess(event) {  
    //Perform your custom success message here  
    alert("Ajax request success!!!!");  
  }  
}
```

```
}
```

```
}
```

```
'
```

AjaxError

The `ajaxError` event is called, if an error occurred with the Ajax request. You can display the customized error details in the event method.

```
`html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
/ajaxError) = "onAjaxFailure($event)">
</bold-reportviewer>
`
```

```
`typescript
```

```
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'Sales Order Detail.rdl';
}
onAjaxFailure(event) {
alert("Status: " + event.status + "\n" +
"Error: " + event.responseText);
```

Print report

[View report in print mode](#)

```
}
```

```
}
```

```
'
```

You can never have both an error and a success callback with a request.

Print report

The Report Viewer provides print report option in the toolbar to print a copy of the report. The Page Setup dialog allows you to set the paper size or other page setup properties. To see print margins, click **Print Layout** on the toolbar.

You can set values in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values for the Page Setup dialog is based on the report properties set in the design view.

[View report in print mode](#)

Print margins are displayed in the Print Layout only. To view the report in print mode by default, set the **printMode** property to true.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"=
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[printMode] = "isPrintMode">
</bold-reportviewer>
'
```

```
'typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
```

```
public serverUrl: string;
public isPrintMode: boolean;
constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
    this.reportPath = 'GroupingAgg.rdl';
    this.isPrintMode = true;
}
}
`
```

By default, the Report Viewer renders the report in normal layout in which the print margins are not displayed.

[Print in new page](#)

To open the print in a new tab of the current browser, set the `printOption` property to `NewTab`. By default, it shows the print dialog in the same page.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[printOption] = "printOption">
</bold-reportviewer>
`

`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
```

Print report

Set page orientation and paper size

```
public serverUrl: string;
public printOption: any;
constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
    this.reportPath = 'GroupingAgg.rdl';
    this.printOption = ej.ReportViewer.PrintOptions.NewTab;
}
}
`
```

The pop-up blocker should be enabled for the page to open the print view in a new tab.

[Set page orientation and paper size](#)

You can specify the print page paper size and orientation at client-side to change the page setup properties by setting the `pageSettings` property.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[printMode] = "isPrintMode"
[pageSettings] = "pageSettings">
</bold-reportviewer>
`

`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
```

```
public serverUrl: string;
public pageSettings: any;
public isPrintMode: boolean;
constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.reportPath = 'GroupingAgg.rdl';
    this.isPrintMode = true;
    this.pageSettings = {
        orientation: ej.ReportViewer.Orientation.Portrait,
        paperSize: ej.ReportViewer.PaperSize.A3,
    };
}
`
```

Set report margin

To set margin values to the report page setup, use the `margins` property and specify the value to top, right, bottom, and left.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[printMode] = "isPrintMode"
[pageSettings] = "pageSettings">
</bold-reportviewer>
`

'typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
    selector: 'ej-app',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']
```

```
)  
export class AppComponent {  
    public serviceUrl: string;  
    public reportPath: string;  
    public serverUrl: string;  
    public pageSettings: any;  
    public isPrintMode: boolean;  
    constructor() {  
        this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';  
        this.reportPath = 'GroupingAgg.rdl';  
        this.isPrintMode = true;  
        this.pageSettings = {  
            margins: {  
                top: 0.5,  
                right: 0.25,  
                bottom: 0.25,  
                left: 0.25  
            }  
        };  
    }  
}  
`
```

The values set in the margin property is considered as inches input.

Set page height and width

To set the height and width values to the report page setup, use the `height` and `width` properties.

```
'html  
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"  
[processingMode] = "Remote"  
[reportServerUrl] = "serverUrl"  
[reportPath] = "reportPath"  
[printMode] = "isPrintMode"  
[pageSettings] = "pageSettings">
```

```
</bold-reportviewer>
`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
  selector: 'ej-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  public serviceUrl: string;
  public reportPath: string;
  public serverUrl: string;
  public pageSettings: any;
  public isPrintMode: boolean;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.reportPath = 'GroupingAgg.rdl';
    this.isPrintMode = true;
    this.pageSettings = {
      height: 2.69,
      width: 28.27
    };
  }
}
`
```

The values set in the height and width properties is considered as inches input.

[Print report with images](#)

When the report has more images, the browser will send the report stream to the print dialog before the images are completely loaded. To load the print report stream with complete images, set the **EmbedImageData** property to true in **OnInitReportOptions** as shown in the following code.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

```
{  
reportOption.ReportModel.EmbedImageData = true;  
}  
,
```

Replace the following code sample in client-side HTML file.

```
`html  
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"  
[processingMode] = "Remote"  
[reportServerUrl] = "serverUrl"  
[reportPath] = "reportPath">  
</bold-reportviewer>  
,
```

```
`typescript  
import { Component, ViewChild } from '@angular/core';  
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';  
@Component({  
selector: 'ej-app',  
templateUrl: './app.component.html',  
styleUrls: ['./app.component.css']  
})  
export class AppComponent {  
public serviceUrl: string;  
public reportPath: string;  
public serverUrl: string;  
constructor() {  
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';  
this.reportPath = 'GroupingAgg.rdl';  
}  
}
```

In this tutorial, the **Product Details.rdl** report is used, and it can be downloaded from [here](#).

External styles in report printing

While printing a report, the external styles used in the application overrides the printable page style and prints output with incorrect alignments. To avoid the external script overriding, set the `isStyleLoad` property to false, which will print the page using only the Report Viewer styles.

`'html'`

```
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[locale]= "Remote"
(reportPrint) = "onReportPrint($event)">
</bold-reportviewer>
```

`'`

`'typescript'`

```
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'Product Details.rdl';
}
onReportPrint(event) {
event.isStyleLoad = false;
}
}
```

Show print progress

The Report Viewer provides events that help you show the progress information, when the printing process takes a long time to complete.

To show print progress, follow these steps:

1. Set the `printProgressChanged` in Report Viewer initialization.
2. Implement the function and add code samples to show a custom message based on the print progress status as shown in the following code snippet.

`html

```
<ej-ReportViewer id="ReportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[locale]= "Remote"
(printProgressChanged) = "onPrintProgressChanged($event)">
</ej-ReportViewer>
```

`

`typescript

```
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
this.reportPath = 'Product Details.rdl';
```

```
}

onPrintProgressChanged(event) {
    if (event.stage === "beginPrint") {
        console.log(event.stage);
        //$('#viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing print data..
        Please wait..." });
    }

    if (event.stage === "printStarted") {
        console.log(event.stage);
        //var popupObj1 = $('#viewer').data('ejWaitingPopup');
        //popupObj1.hide();
    }

    else if (event.stage === "preparation") {
        console.log(event.stage);
        if (event.preparationStage === "dataPreparation") {
            console.log(event.preparationStage);
            console.log(event.totalPages);
            console.log(event.currentPage);

            //if (event.totalPages > 1 && event.currentPage > 1) {
            //    var progressPercentage = Math.floor((event.currentPage / event.totalPages) * 100);
            //    if (progressPercentage > 0) {
            //        var popupObj2 = $('#viewer').data('ejWaitingPopup');
            //        popupObj2.setModel({ text: "Preparing print data.." + progressPercentage + " % completed..
            Please wait..." });

            //}
            //}
        }
    }

    event.handled = true;
}
}
```

Remove empty spaces in printing

The extra blank page is created when the body of your report is too wide for your page. To make the report appear on a single page, all the content within the report body must fit on the physical page, and the body width should be as the following formula:

Body Width <= Page Width - (Left Margin + Right Margin)

For more details to remove the empty pages in the report while designing, refer to the knowledge base article of [report page sizing](#).

Export report

The Report Viewer provides events and properties to control and customize the report exporting functionality.

Export event handling

You can show the progress information, when the exporting process takes long time to complete using the `exportProgressChanged` event.

1. Set the `exportProgressChanged` event in Report Viewer initialization.
2. Implement the function and replace the following code samples to get the log message based on the progress stage.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
(exportProgressChanged) = "onExportProgressChanged($event)">
</bold-reportviewer>
`
```

```
'typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
  selector: 'ej-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
```

```
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
    this.reportPath = 'GroupingAgg.rdl';
}
onExportProgressChanged(event) {
    if (event.stage === "beginExport") {
        console.log(event.stage);
    }
    else if (event.stage === "exportStarted") {
        console.log(event.stage);
    }
    else if (event.stage === "preparation") {
        console.log(event.stage);
        console.log(event.format);
        console.log(event.preparationStage);
    }
    event.handled = true;
}
}
```

Export data visualization items

To export the reports with data visualization components, it is mandatory to configure the web scripts in Report Viewer Web API controller. If the report definition uses chart, gauge and map report items then configure the scripts in Web API as the following steps,

1. Open the Report Viewer Web API controller.
2. Configure the below scripts and styles in `OnInitReportOptions` on Web API controller.
 - o `jquery-1.10.2.min.js`
 - o `bold.reports.common.min.js`
 - o `bold.reports.widgets.min.js`
 - o `ej.chart.min.js`
 - o `ej2-base.min.js`
 - o `ej2-data.min.js`

- ej2-pdf-export.min.js
- ej2-svg-base.min.js
- ej2-lineargauge.min.js
- ej2-circulargauge.min.js
- ej.map.min.js
- bold.report-viewer.min.js

3. You can replace the following codes in Report Viewer Web API controller.

`csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
        //Gauge component scripts
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
        //Map component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
        //Report Viewer Script
        "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
    };
    reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
    {
        "https://code.jquery.com/jquery-1.10.2.min.js"
    };
}
```

The data visualization components will not export without the above script configurations.

Export data visualization items in azure environment

Report Viewer uses WebBrowser to export the data visualization items to PDF, Word, Excel file formats. The WebBrowser is not supported in azure environment. To overcome this limitation in azure environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

To download PhantomJS application and deploy it on your machine, you should accept it's license terms on [LICENSE](#) and [Third-Party](#) document.

1. Download PhantomJS from [here](#) and extract the download file.
2. Copy the PhantomJS.exe file from the extracted bin folder and paste into wwwroot/PhantomJS folder in your application.
3. Open the Report Viewer Web API controller.
4. Set the UsePhantomJS property to true and PhantomJSPath property in OnInitReportOptions method.

`csharp

```
// Method will be called to initialize the report information to load the report with ReportHelper for processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    reportOption.ReportModel.ExportResources.UsePhantomJS = true;
    reportOption.ReportModel.ExportResources.PhantomJSPath = basePath + @"\PhantomJS\";
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
        //Gauge component scripts
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
    }
}
```

```
//Map component script  
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",  
//Report Viewer Script  
"https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",  
};  
reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>  
{  
    "https://code.jquery.com/jquery-1.10.2.min.js"  
};  
}  
`
```

The **Scripts** and **Dependent scripts** must be added to export the items. For more details refer to the [export data visualization items](#) section.

Change Excel and Word export format

You can change the default file format to any other file format using the **excelFormat** and **wordFormat** properties. The following code sample changes the default versions.

```
'html  
  
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"  
[processingMode] = "Remote"  
[reportServerUrl] = "serverUrl"  
[reportPath] = "reportPath"  
[exportSettings] = "exportSettings">  
</bold-reportviewer>  
`  
  
'typescript  
  
import { Component, ViewChild } from '@angular/core';  
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';  
@Component({  
    selector: 'ej-app',  
    templateUrl: './app.component.html',  
    styleUrls: ['./app.component.css']  
})  
export class AppComponent {
```

Export report

Hide specific export type for report

```
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public exportSettings: any;
constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
    this.reportPath = 'GroupingAgg.rdl';
    this.exportSettings = {
        excelFormat: ej.ReportViewer.ExcelFormats.Excel2013,
        wordFormat: ej.ReportViewer.WordFormats.Word2013
    }
}
}
}
`
```

Hide specific export type for report

Show or hide the default export types available in the component using the `exportOptions` property. The following code hides the HTML export type from the default export options.

```
`html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[exportSettings] = "exportSettings">
</bold-reportviewer>
`  
  
`typescript
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
@Component({
    selector: 'ej-app',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']
})
```

```
export class AppComponent {  
    public serviceUrl: string;  
    public reportPath: string;  
    public serverUrl: string;  
    public exportSettings: any;  
    constructor() {  
        this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';  
        this.reportPath = 'GroupingAgg.rdl';  
        this.exportSettings = {  
            exportOptions: ej.ReportViewer.ExportOptions.All & ~ej.ReportViewer.ExportOptions.HTML &  
            ~ej.ReportViewer.ExportOptions.Word  
        }  
    }  
}
```

PDF export options

The **PDFOptions** provides properties to manage PDF export behaviors. You should set the properties in the **OnInitReportOptions** method of the Web API service.

Export with complex scripts

To export reports with the complex scripts, set the **ComplexScript** property of **PDFOptions** instance to true.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()  
    {  
        EnableComplexScript = true  
    };  
}
```

PDF conformance

You can export the report as a **PDF/A-1b** document by specifying the **PdfConformanceLevel.Pdf_A1B** conformance level in the **PdfConformanceLevel** property.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

```
{  
reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()  
{  
PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B  
};  
}  
'
```

Add custom PDF fonts

You can add custom fonts to the PDF exported document by adding the font streams to **Fonts** collection in **PDFOptions** instance.

To add custom fonts to the PDF exported document, follow these steps:

1. Add the font **.ttf** files into your application **App_Data** folder.
2. In the Solution Explorer, open the properties of the font file and set the property **Copy to Output Directory** as **Copy always**.
3. Initialize the **Font** collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()  
{  
//Load Missing font stream  
Fonts = new Dictionary<string, System.IO.Stream>  
{  
{ "Segoe UI",  
System.IO.File.OpenRead(System.Web.Hosting.HostingEnvironment.MapPath(@"~/AppData/fontsymbols.ttf")) }  
}  
};  
}'
```

If any fonts used in the report definition is not installed or available in the local system, then you should load the font stream. In the above code, **font_symbols** font stream is loaded to export the **Sales Order Detail.rdl** report as symbols.

Word export options

The **WordOptions** provides properties to manage Word document export behaviors.

Word document type

You can save the report to the required document version by setting the **FormatType** property.

```
'csharp
```

```
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
{  
    FormatType = BoldReports.Writer.WordFormatType.Docx  
};  
'
```

Word document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells and render the word document elements without nested grid layout by setting the **LayoutOption** to **TopLevel**. The **ParagraphSpacing** is the distance value added between two elements in the document.

```
'csharp
```

```
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
{  
    LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel,  
    ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()  
    {  
        Bottom = 0.5f,  
        Top = 0.5f  
    }  
};  
'
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, added an empty paragraph between two tables.

Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without password.

The following are the types of protection:

- **AllowOnlyComments**: Adds or modifies only the comments in the Word document.
- **AllowOnlyFormFields**: Modifies the form field values in the Word document.
- **AllowOnlyRevisions**: Accepts or rejects the revisions in the Word document.

- **AllowOnlyReading:** Allows you to view the content only in the Word document.
- **NoProtection:** Accesses or edits the Word document contents as normally.

```
'csharp
```

```
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
{  
    ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading  
};  
'
```

Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013  
};  
'
```

Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the **LayoutOption** to **IgnoreCellMerge**.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge  
};  
'
```

Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the **ExcelSheetProtection** or enabling the **ReadOnlyRecommended** properties.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    ReadOnlyRecommended = true,  
'
```

```
ExcelSheetProtection = ExcelSheetProtection.DeletingColumns  
};  
`
```

PowerPoint export options

You can save the report to the required PowerPoint version by setting the `FormatType` property.

```
`csharp  
reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()  
{  
    FormatType = BoldReports.Writer.PPTSaveType.PowerPoint2013  
};  
`
```

CSV export options

The `CsvOptions` allows you change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
`csharp  
reportOption.ReportModel.CsvOptions = new BoldReports.Writer.CsvOptions()  
{  
    Encoding = System.Text.Encoding.Default,  
    FieldDelimiter = ",",  
    UseFormattedValues = false,  
    Qualifier = "#",  
    RecordDelimiter = "@",  
    SuppressLineBreaks = true,  
    FileExtension = ".txt"  
};  
`
```

HTML export options

You can hide the separator added at the end of each page by setting the `HidePageSeparator` property to true.

```
`csharp  
reportOption.ReportModel.HTMLOptions = new BoldReports.Writer.HTMLOptions()  
{  
    HidePageSeparator = true  
};
```

Password protect exported document

Allows you protect the exported document such as PDF, Microsoft Word, Microsoft Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet demonstrates how to encrypt the exported document with user defined password.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //PDF encryption
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions();
    reportOption.ReportModel.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
    {
        UserPassword = "password"
    };
    //Word encryption
    reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
    {
        EncryptionPassword = "password"
    };
    //Excel encryption
    reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
    {
        PasswordToModify = "password",
        PasswordToOpen = "password"
    };
    //PPT encryption
    reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()
    {
        EncryptionPassword = "password"
    };
}
```

>Password protection is not supported for HTML export format.

Toolbar customization

You can hide the component toolbar to show customized user interface or to customize the toolbar icons and element's appearances using the templates and Report Viewer toolbar customization properties.

In this tutorial, the Sales Order Detail.rdl report is used, and it can be downloaded from [here](#). You can add the reports from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

Hide parameter block and toolbar items

To hide toolbar items, set the `toolbarSettings` property. The following code can be used to remove the parameter option from the toolbar and hide the parameter block.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[toolbarSettings] = "toolbarSettings">
</bold-reportviewer>
'

'ts
import { Component, ViewChild } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public toolbarSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.toolbarSettings = {
```

```
items: ~ej.ReportViewer.ToolbarItems.Parameters,  
}  
}  
}  
`
```

The following code sample hides the print options from the toolbar items.

```
`html  
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"  
[processingMode] = "Remote"  
[reportServerUrl] = "serverUrl"  
[reportPath] = "reportPath"  
[toolbarSettings] = "toolbarSettings">  
</bold-reportviewer>  
`  
  
'ts  
import { Component, ViewChild } from '@angular/core';  
@Component({  
selector: 'ej-app',  
templateUrl: './app.component.html',  
styleUrls: ['./app.component.css']  
})  
export class AppComponent {  
public serviceUrl: string;  
public reportPath: string;  
public serverUrl: string;  
public toolbarSettings: any;  
constructor() {  
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';  
this.reportPath = 'GroupingAgg.rdl';  
this.toolbarSettings = {  
items: ~ej.ReportViewer.ToolbarItems.Print,  
}
```

```
}
```

```
}
```

```
'
```

Similarly, you can show or hide all other toolbar options with the help of `toolbarSettings.items` enum.

[Enable stop option in toolbar](#)

To enable stop option in toolbar, set the `toolbarSettings.items` property to `ej.ReportViewer.ToolbarItems.All`. The following code can be used to enable stop option in toolbar.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[toolbarSettings] = "toolbarSettings">
</bold-reportviewer>
'

'ts
import { Component, ViewChild } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public toolbarSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.toolbarSettings = {
items: ej.ReportViewer.ToolbarItems.All,
```

```
}
```

```
}
```

```
}
```

```
,
```

Hide toolbar

To hide the Report Viewer toolbar, set the `showToolbar` property to false.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[toolbarSettings] = "toolbarSettings">
</bold-reportviewer>
```

```
,
```

```
'ts
```

```
import { Component, ViewChild } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public toolbarSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.toolbarSettings = {
showToolbar: false,
}
}
```

```
}
```

```
}
```

```
'
```

Decide or hide the export option

The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. The following code hides the HTML export type from the default export options.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[exportSettings] = "exportSettings">
</bold-reportviewer>
'

`ts
import { Component, ViewChild } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public exportSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.exportSettings = {
exportOptions: ej.ReportViewer.ExportOptions.All & ~ej.ReportViewer.ExportOptions.HTML
}
}
```

```
}
```

```
}
```

```
'
```

Add custom items to the export drop-down

To add custom items to the export drop-down available in the Report Viewer toolbar, use the `customItems` property and provide the JSON array of collection input with the `index`, `cssClass` name, and `value` properties. Register the `exportItemClick` event to handle the custom item actions as given in following code snippet.

```
'html
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[exportSettings] = "exportSettings"
(exportItemClick) = "onExportItemClick($event)">
</bold-reportviewer>
'

`ts
import { Component, ViewChild } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public exportSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.exportSettings = {
```

```
customItems: [{  
    index: 2,  
    cssClass: "",  
    value: 'Text File'  
},  
{  
    index: 4,  
    cssClass: "",  
    value: 'DOT'  
}]  
};  
}  
  
//Export click event handler  
onExportItemClick(event) {  
if (event.value === "Text File") {  
    //Implement the code to export report as Text  
    alert("Text File export option clicked");  
} else if (event.value === "DOT") {  
    //Implement the code to export report as DOT  
    alert("DOT export option clicked");  
}  
}  
}  
}  
`
```

Add custom toolbar item

You can add custom items to Report Viewer toolbar using the `toolbarSettings` property. You should register the `toolBarItemClick` event to handle the newly added custom items action.

Add custom item to exiting toolbar group

To add a custom item to existing toolbar group use the `customItems` property in `toolbarSettings` and provide the JSON array of collection input with the `groupIndex`, `index`, `itemType`, `cssClass` name, and `tooltip` properties as given in following code snippet.

```
'html  
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"
```

```
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[toolbarSettings] = "toolbarSettings"
(toolBarItemClick) = "onToolBarItemClick($event)">
</bold-reportviewer>
`  

'ts
import { Component, ViewChild } from '@angular/core';
@Component({
selector: 'ej-app',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public toolbarSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.toolbarSettings = {
showToolbar: true,
items: ~ej.ReportViewer.ToolbarItems.Print,
customItems: [{  

groupIndex: 1,  

index: 1,  

type: 'Default',  

cssClass: "e-icon e-mail e-reportviewer-icon",  

id: 'E-Mail',  

tooltip: {  

header: 'E-Mail',
```

```
content: 'Send rendered report as mail attachment'  
}  
}  
}  
}  
}  
  
//Toolbar click event handler  
onToolBarItemClick(event) {  
if (event.value === "CustomItem") {  
//Implement the code to CustomItem toolbar option  
alert("CustomItem toolbar option Clicked");  
}  
}  
}  
`
```

Add new toolbar group

To add a new toolbar group and custom items to it, use the `customGroups` property in the `toolbarSettings` and provide the JSON array of collection input with the `groupIndex` and `items` properties. The `items` should have the `itemType`, `cssClass`, and `tooltip` properties as given in following code snippet.

```
'html  
  
<bold-reportviewer id="reportViewer_Control"  
[reportServiceUrl] = "serviceUrl"  
[processingMode] = "Remote"  
[reportServerUrl] = "serverUrl"  
[reportPath] = "reportPath"  
[toolbarSettings] = "toolbarSettings"  
(toolBarItemClick) = "onToolBarItemClick($event)">  
</bold-reportviewer>  
`  
  
'ts  
  
import { Component, ViewChild } from '@angular/core';  
@Component({  
selector: 'ej-app',  
templateUrl: './app.component.html',
```

```
styleUrls: ['./app.component.css']
})

export class AppComponent {
public serviceUrl: string;
public reportPath: string;
public serverUrl: string;
public toolbarSettings: any;
constructor() {
this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
this.reportPath = 'GroupingAgg.rdl';
this.toolbarSettings = {
showToolbar: true,
items: ~ej.ReportViewer.ToolbarItems.Print,
customGroups: [{  
    items: [{  
        type: 'Default',
cssClass: "e-icon e-mail e-reportviewer-icon CustomGroup",
id: 'CustomGroup',
tooltip: { header: 'CustomGroup', content: 'toolbargroups' }
},
{
type: 'Default',
cssClass: "e-icon e-mail e-reportviewer-icon subCustomGroup",
id: 'subCustomGroup',
tooltip: { header: 'subCustomGroup', content: 'subtoolbargroups' }
}],
groupIndex: 3
}]
}
}
}

//Toolbar click event handler
onToolBarItemClick(event) {
if (event.value === "CustomGroup") {
```

| | |
|----------------|--------------------------------------|
| Custom actions | Send a report as an email attachment |
|----------------|--------------------------------------|

```
//Implement the code to CustomGroup toolbar option
alert("CustomGroup toolbar option clicked");

}

if (event.value === "subCustomGroup") {
//Implement the code to subCustomGroup toolbar option
alert("SubCustomGroup toolbar option clicked");

}
}

}

`
```

Custom actions

This section explains you the steps required to add user defined buttons in Report Viewer toolbar and invoke custom actions.

[Send a report as an email attachment](#)

This topic describes steps required to create custom email option and share a report as an email attachment to other users.

Add email button in Report Viewer

1. Create an email button in the toolbar using the `customItems` property with the values such as `groupIndex`, `index`, `itemType`, `cssClass`, and `tooltip`. The `toolBarItemClick` event triggers when you click the email button.
2. Access the Report Viewer model and create a JSON array for sending requests to the Web API server. You can use the following codes to create an event with custom action.

```
'js
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/ReportsWebApi",
        reportPath: '~/App_Data/Sales Order Detail.rdl',
        toolbarSettings: {
            showToolbar: true,
            items: ej.ReportViewer.ToolbarItems.All & ~ej.ReportViewer.ToolbarItems.Print,
            customItems: [{
                groupIndex: 1,
                index: 1,
```

```
type: 'Default',
cssClass: "e-icon e-mail e-reportviewer-icon",
id: 'E-Mail',
tooltip: {
header: 'E-Mail',
content: 'Send rendered report as mail attachment'
}
}]
},
toolBarItemClick: 'ontoolBarItemClick'
});
});
//Toolbar click event handler
function ontoolBarItemClick(args) {
if (args.value == "E-Mail") {
var proxy = $('#viewer').data('boldReportViewer');
var Report = proxy.model.reportPath;
var lastsIndex = Report.lastIndexOf("/");
var reportName = Report.substring(lastsIndex + 1);
var requrl = proxy.model.reportServiceUrl + '/SendEmail';
var _json = {
exportType: "PDF", reportViewerToken: proxy._reportViewerToken, ReportName: reportName
};
$.ajax({
type: "POST",
contentType: "application/json; charset=utf-8",
url: requrl,
data: JSON.stringify(_json),
dataType: "json",
crossDomain: true
})
}
}
```

```
</script>
```

Create custom email action

To create custom email action, follow these steps:

1. Create a new action method `SendEmail` in the Web API service.
2. Export the report to the required type using the `ReportHelper.GetReport` method to send a report stream as an attachment.

The following code sample exports the report to stream and send it as an attachment to a specified mail address. In the code, the `SmtpClient` method is used to send the report as an email attachment.

```
'csharp
```

```
public object SendEmail(Dictionary<string, object> jsonResult)
{
    string _token = jsonResult["reportViewerToken"].ToString();
    var stream = ReportHelper.GetReport(_token, jsonResult["exportType"].ToString());
    stream.Position = 0;
    if (!ComposeEmail(stream, jsonResult["reportName"].ToString()))
    {
        return "Mail not sent !!!";
    }
    return "Mail Sent !!!";
}

public bool ComposeEmail(Stream stream, string reportName)
{
    try
    {
        MailMessage mail = new MailMessage();
        SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");
        mail.IsBodyHtml = true;
        mail.From = new MailAddress("xx@gmail.com");
        mail.To.Add("xx@gmail.com");
        mail.Subject = "Report Name : " + reportName;
        stream.Position = 0;
        if (stream != null)
```

```
{  
    ContentType ct = new ContentType();  
    ct.Name = reportName + DateTime.Now.ToString() + ".pdf";  
    System.Net.Mail.Attachment attachment = new System.Net.Mail.Attachment(stream, ct);  
    mail.Attachments.Add(attachment);  
}  
  
SmtpServer.Port = 587;  
SmtpServer.Credentials = new System.Net.NetworkCredential("xx@gmail.com", "xx");  
SmtpServer.EnableSsl = true;  
SmtpServer.Send(mail);  
return true;  
}  
  
catch (Exception ex)  
{  
    return ex.ToString();  
}  
return false;  
}  
`
```

In the above code sample, the report is exported to PDF format and sent to users using the `SmptClient` method.

3. Build and run the application.

Localization of Bold Reports Angular Report Viewer

Localization of Angular Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. To render the static text with specific culture, refer to the following corresponding culture script files and set culture name to the `locale` property of the Report Viewer.

- `ej.localetexts.fr-FR.min.js`
- `ej.culture.fr-FR.min.js`
- Run the below command, to install the `@boldreports/global` package.

```
'typescript  
npm install @boldreports/global --save  
'
```

- Refer to the `ej.localetexts.fr-FR.min.js` and `ej.culture.fr-FR.min.js` script files from `node_modules` in the `app.module.ts` file .

```
'typescript
```

```
import { NgModule, enableProdMode, ErrorHandler } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { RouterModule } from '@angular/router';
import { FormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/http';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components';
import '../../../../../node_modules/syncfusion-ej-global/i10n/ej.localetexts.fr-FR.min.js';
import '../../../../../node_modules/syncfusion-ej-global/i18n/ej.culture.fr-FR.min.js';
import { AppComponent } from './app.component';
import { TextboxComponent } from './textbox/textbox.component';
...
...
`
```

Note: If you import the culture before the `BoldReportsAngularModule`, you will get the following error in the application. So, you should import the culture after the `@boldreports/angular-reporting-components` package.

![Angular culture error](/static/assets/angular/report-viewer/images/error/cultureerror.png)

Initialize the `locale` property in the `component.ts` and `component.html` pages.

```
'html
```

```
<bold-reportviewer id="reportViewer_Control"
[reportServiceUrl] = "serviceUrl"
[processingMode] = "Remote"
[reportServerUrl] = "serverUrl"
[reportPath] = "reportPath"
[locale] = 'locale'>
</bold-reportviewer>
```

```
'typescript
```

```
import { Component, ViewChild } from '@angular/core';
import { BoldReportsAngularModule } from '@boldreports/angular-reporting-components/src/core';
```

```
@Component({
  selector: 'ej-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})

export class AppComponent {
  public serviceUrl: string;
  public reportPath: string;
  public serverUrl: string;
  public locale: string;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportApi';
    this.reportPath = 'Sales Order Detail.rdl';
    this.locale = "fr-FR";
  }
}
```

Responsive layout rendering of Angular Report Viewer

Report Viewer will adaptively render itself with optimal user interfaces for phone, tablet, or desktop form factors. This helps your application to scale elegantly on all form factors with ease. You can enable responsive layout rendering in Report Viewer by setting `isResponsive` property to true.

```
'typescript

import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})

export class AppComponent {
  title = 'jsreport-sample';
  public serviceUrl: string;
  public reportPath: string;
  public isResponsive: boolean;
```

| | |
|-------------|---------------|
| Limitations | Normal layout |
|-------------|---------------|

```

constructor() {
  this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
  this.reportPath = '~/Resources/docs/sales-order-detail.rdl';
  this.isResponsive = true;
}
}

`js
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [reportPath] =
"reportPath" [isResponsive] = "isResponsive" style="width: 100%;height: 950px">
</bold-reportviewer>
`
```

Normal layout

The following output shows the normal layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in normal layout](/static/assets/angular/report-viewer/images/responsive-layout/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/angular/report-viewer/images/responsive-layout/responsive-layout.png)

Limitations

RDL specification

The Report Viewer control does not support RDL Specification for SQL Server 2000 and SQL Server 2005.

Report layout

1. Vertical alignment in the text box report item is not supported in web rendering.
2. In the Tablix cell split layout process, the entire cell moves to the next page to display the complete cell items, when the table cell width value exceeds the page width.

Expressions

The object function and VB function do not have complete support.

SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you should specify the data source credentials for each report data source to establish database connection.

Samples and demos

Browse and explore the ready-to-use RDL, RDLC reports, samples, online, and offline demos.

Locally installed reports

You can obtain sample rdl and rdlc files from Bold Reports installed location

`%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.`

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view the Angular Report Viewer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view the GitHub Report Viewer demo samples.

Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of Angular Report Viewer application:

Server-side migration

1. In the Solution Explorer, right-click the **References** and remove the `Syncfusion.EJ.ReportViewer` assembly reference.
2. Add the assembly from the Bold Reports NuGet package `BoldReports.Web`. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for `BoldReports.Web` NuGet package, and then install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Web API Controller

1. The `IReportController` interface is moved to `BoldReports.Web.ReportViewer`. Open the Report Viewer Web API Controller file and remove the following using statement.

```
'csharp
using Syncfusion.EJ.ReportViewer;
'
```

2. Add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Your application is successfully upgraded to the latest version of Report Viewer, and you can run the application with new assemblies.

Report export configuration

Now, the **BoldReports.Web** can export the reports with data visualization components only using web components. It is mandatory to configure the web scripts in Report Viewer Web API controller for exporting data visualization components such as chart, gauge, and map that are used in report definition. To configure the scripts in Web API, refer to the following steps:

1. Open the Report Viewer Web API controller.
2. Configure the following scripts and styles in **OnInitReportOptions** on Web API controller:
 - o jquery-1.10.2.min.js
 - o bold.reports.common.min.js
 - o bold.reports.widgets.min.js
 - o ej.chart.min.js - Exports the chart item. Add this script only if your report contains the chart report item.
 - o ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js - Exports the gauge item. Add this script only if your report contains the gauge report item.
 - o ej.map.min.js - Exports the map item. Add this script only if your report contains the map report item.
 - o bold.report-viewer.min.js
3. Replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        //Gauge component scripts
        resourcesPath + @"\bold-reports\common\ej2-base.min.js",
        resourcesPath + @"\bold-reports\common\ej2-data.min.js",
    }
}
```

```
resourcesPath + @"\"bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\"bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\"bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\"bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\"bold-reports\data-visualization\ej.map.min.js",
//Report Viewer Script
resourcesPath + @"\"bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\"bold-reports\bolt.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\"jquery-1.7.1.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

NuGet Packages for Angular

Refer to the following steps to configure Bold Reporting NuGet packages for Angular Web API application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

1. Open Visual Studio application.
2. On the **Tools** menu, select **Options**.
3. Expand the **NuGet Package Manager** and select **Package Sources**.
4. Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: [NuGet.org](#)

Source: <https://api.nuget.org/v3/index.json>

![Online NuGet Configure] (/static/assets/angular/report-viewer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reports NuGet packages are shipped into our Bold Reporting Tools build. To configure the packages from Bold Reports installed location, use the following steps:

1. Open your Visual Studio application.
2. On the **Tools** menu, select **Options**.
3. Expand the **NuGet Package Manager**, and then select **Package Sources**.
4. Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages.

![Offline NuGet Configure](/static/assets/angular/report-viewer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

Installing NuGet packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

1. On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**. Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.
2. By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages **BoldReports.Web**, and then click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

1. On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.
2. Run the following NuGet installation commands:

```
'cmd
```

install specified package in default project

```
Install-Package <Package Name>
```

install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

For example:

'cmd

install specified package in default project

Install-Package BoldReports.Web

install specified package in default project with specified Package Source

Install-Package BoldReports.Web -Source "https://api.nuget.org/v3/index.json"

install specified package in specified project

Install-Package BoldReports.Web -ProjectName BoldReportsApplication

'

Upgrading NuGet packages**Upgrading using NuGet Package Manager**

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

1. On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....** Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.
2. Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

Upgrading using Package Manager Console

To update the installed Bold Reports NuGet packages using the Package Manager Console:

1. On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.
2. Run the following NuGet installation commands:

'cmd

Update specific NuGet package in default project

Update-Package <Package Name>

Update all the packages in default project

Update-Package

Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

Update specified Bold Reports NuGet package

Update-Package BoldReports.Web

Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportsApplication

Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

1. Download the latest [NuGet CLI](#).

To update the existing nuget.exe to latest version use the following command:

`cmd

nuget update -self

2. Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reports NuGet packages.

`cmd

update all NuGet packages from config file

nuget update <configPath> [options]

update all NuGet packages from specified Packages Source

nuget update -Source <Source Location> [optional]

configPath is optional. It identifies the **package.config** or solutions file lists the packages utilized in the project.

For example:

```
'cmd
```

Update all NuGet packages from config file

```
nuget update "C:\Users\BoldReportsApplication\package.config"
```

Update all NuGet packages from specified Packages Source

```
nuget update -Source "https://api.nuget.org/v3/index.json"
```

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

Deployment

This topic explains about simplest production-ready deployment of your Angular Report Viewer application to a remote server by creating a production build.

To create a production build run the following command and copy the output directory to a web server.

```
'typescript
```

```
node_modules\.bin\ng build --prod
```

Please refer to the [angular deployment](#) documentation to know more about deploying your Angular application.

If you get error `FATAL ERROR: CALLANDRETRYLAST Allocation failed - JavaScript heap out of memory add --maxoldspace=xx space in ngc.cmd and ng.cmd file from node_modules\.bin folder.`

Modify `ngc.cmd`

```
'js
```

```
@IF EXIST "%~dp0\node.exe" (
    "%~dp0\node.exe" --maxoldspace_size=8192 "%~dp0..\@angular\compiler-cli\src\main.js" %*
) ELSE (
    @SETLOCAL
    @SET PATHEXT=%PATHEXT%;.JS;=%
    node --maxoldspace_size=8192 "%~dp0..\@angular\compiler-cli\src\main.js" %*
)
```

Modify `ng.cmd`

```
'js
```

```
@IF EXIST "%~dp0\nginx.exe" (
    "%~dp0\nginx.exe" --maxoldspace_size=8192 "%~dp0..\@angular\cli\bin\ng" %*
) ELSE (
    @SETLOCAL
    @SET PATHEXT=%PATHEXT%;.JS;=%
    node --maxoldspace_size=8192 "%~dp0..\@angular\cli\bin\ng" %*
)
```

Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports Angular Report Viewer.

1. [How can improve the performance and handle the large amounts of data with Report Viewer?](#)
2. [Is Report Viewer compatible with latest version of JQuery library?](#)
3. [Is the back action from drillthrough report will load the report again with Report Viewer?](#)
4. [Is possible to change the culture of the date time parameter?](#)
5. [Is it possible to hide report parameters?](#)
6. [Is it possible to hide the export options in Report Viewer?](#)
7. [Is it possible to load reports providing parameter values at runtime?](#)

Is possible to change the culture of the date time parameter

Yes, we can change the culture of the date time parameter for Report Viewer by changing the locale. You can make use the following reference to change the locale of Report Viewer.

[ReportViewer Localization](#)

In Report Viewer, we could not change the culture of specific parameter or UI. So, we have to achieve the requirements only by changing the culture.

Is it possible to hide the parameters in Report Viewer

Yes, it is possible to hide the parameters in Report Viewer. On hiding the parameters, the users will not be able to have the parameter block in the Report Viewer. Refer to this [Hide parameter block and toolbar items](#) section.

Is it possible to hide the export options in Report Viewer

Yes, it is possible to hide the export options in Report Viewer. The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. Refer to this [Decide or Hide the export options](#) section.

Is it possible to load reports providing parameter values at runtime

Yes, it is possible to load reports with application inputs as parameters in Report Viewer. You need to provide the input values to the Report Viewer from client side at runtime using the `parameters` property, which accepts JSON array values. Refer to this [Set parameter at client](#) section.

Angular Report Viewer Reporting Service

The Angular Report Viewer requires a Web API service to process the report files. The following topics explains how to create reporting Web API service to preview the reports.

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

IReportController

The `IReportController` interface has the declaration of action methods that is defined in the Web API Controller for processing the RDL, RDLC, and SSRS report and handling the request from the Report Viewer control. The `IReportController` has the following action methods declaration.

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

'csharp

```
public class ReportsController: ApiController,IReportController
{
    /// <summary>
    /// Action (HttpGet) method for getting resource for report.
    /// </summary>
    /// <param name="key">The unique key to get the required resource.</param>
    /// <param name="resourceType">The type of the requested resource.</param>
    /// <param name="isPrinting">If set to <see langword="true"/>, then the resource is generated for printing.</param>
    /// <returns>The object data.</returns>
    public object GetResource(string key, string resourceType, bool isPrinting)
    {
        //Returns the report resource for the requested key.
        return ReportHelper.GetResource(key, resourceType, isPrinting);
    }
    /// <summary>
    /// Report Initialization method that is triggered when report begin processed.
    /// </summary>
```

```
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOptions)
{
    //You can update report options here
}

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOptions)
{
    //You can update report options here
}

/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
    //Processes the report request and returns the result.
    return ReportHelper.ProcessReport(jsonData, this);
}
}
```

Create ASP.NET Web API service

In this section, you will learn how to create a Web API Service for Report Viewer using the new ASP.NET Empty Web Application template.

1. Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.
2. Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.
3. Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![ASP.NET Web Application project template](/static/assets/angular/report-viewer/images/report-service/aspnetmvc5-template.png)

4. Choose **Empty, Web API** and then click **OK**. Now, the Web application project is created with Web API.

![Select Web API and Empty options](/static/assets/angular/report-viewer/images/report-service/add-web-api.png)

Configure Report Viewer Web API

1. Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

2. Search for **BoldReports.Web** NuGet packages, and install them in your Web application.

Package | Purpose

PostReportAction | Action (HttpPost) method for posting the request in report process.

OnInitReportOptions | Report initialization method that occurs when the report is about to be processed.

OnReportLoaded | Report loaded method that occurs when the report and sub report start loading.

GetResource | Action (HttpGet) method to get resource for the report.

ReportHelper

The class **ReportHelper** contains helper methods that help to process a Post or Get request from the Report Viewer control and return the response to the Report Viewer control. It has the following methods:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

1. Right-click **Controller** folder in your project and select **Add > New Item** from the context menu.
2. Select **Web API Controller Class** from the listed templates and name it as **ReportViewController.cs**

![Provide controller name](/static/assets/angular/report-viewer/images/report-service/add-web-api-controller.png)

3. Click **Add**.

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

4. Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

5. Inherit the **IReportController** interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
public class ReportViewerController : ApiController, IReportController
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]
    public object GetResource(string key, string resourcetype, bool isPrint)
    {
        return ReportHelper.GetResource(key, resourcetype, isPrint);
    }

    // Method that will be called when initialize the report options before start processing the report
    public void OnInitReportOptions(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }

    // Method that will be called when reported is loaded
    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }
}
```

}

,

Add routing information

1. To configure routing to include an action name in the URI, open the `WebApiConfig.cs` file and change the `routeTemplate` in the `Register` method as follows,

```
'csharp
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Web API configuration and services
        // Web API routes
        config.MapHttpAttributeRoutes();
        config.Routes.MapHttpRoute(
            name: "DefaultApi",
            routeTemplate: "api/{controller}/{action}/{id}",
            defaults: new { id = RouteParameter.Optional }
        );
    }
}
```

,

2. Compile and run the Web API service application.

Enable Cross-Origin Requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

1. Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.
2. Search for `Microsoft.AspNet.WebApi.Cors` NuGet packages, and install them in your Web API application.
3. Call `EnableCors` in `WebApiConfig` to add CORS services to the `Register` method. Replace the following code to allow any origin requests.

```
`csharp
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Add Enable Cors
        config.EnableCors();

        // Web API configuration and services
        // Web API routes
        config.MapHttpAttributeRoutes();

        config.Routes.MapHttpRoute(
            name: "DefaultApi",
            routeTemplate: "api/{controller}/{action}/{id}",
            defaults: new { id = RouteParameter.Optional }
        );
    }
}
`
```

4. To specify the CORS policy for API controller, add the [EnableCors] attribute to the controller class. Specify the policy name.

```
`csharp
[EnableCors(origins: "", headers: "", methods: "*")]
public class ReportViewerController : ApiController, IReportController
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]

    public object GetResource(string key, string resourcetype, bool isPrint)
```

```
{  
    return ReportHelper.GetResource(key, resourcetype, isPrint);  
}  
  
// Method that will be called when initialize the report options before start processing the report  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    // You can update report options here  
}  
  
// Method that will be called when reported is loaded  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    // You can update report options here  
}  
}
```

Create ASP.NET Core Web API Service

To create an ASP.NET Core Web API for Report Viewer using a new ASP.NET Core Web Application template, follow these steps:

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Viewer API creation.

1. Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.
2. Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.
3. Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/angular/report-viewer/images/report-service/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. Right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages** and then search for **BoldReports.Net.Core** package, and install to the application. The following provides detail of the packages and its usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

Newtonsoft.Json | Serialize and deserialize the data for report viewer. It is a mandatory package for the report viewer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer. It should be referred in project when renders the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0.

Configure Web API

The interface **IReportController** has declaration of action methods that are defined in the Web API Controller for processing the RDL, RDLC, and SSRS reports and for handling request from the Report Viewer control. The **IReportController** has the following action methods declaration:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

1. Right-click the project and select **Add > New Item** from the context menu.
2. In the Add New Item dialog, select **API Controller** class and name it as **ReportViewerController.cs**

![Adding a new controller to the project](/static/assets/angular/report-viewer/images/report-service/add-aspnet-core-api-controller.png)

3. Click **Add**.

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

4. Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

5. Inherit the **IReportController** interface, and then implement its methods.

6. Create local references for the interfaces given in following table.

Interface | Purpose

IMemoryCache | Report Viewer requires a memory cache to store the information of consecutive client request and have the rendered report viewer information in server.

IHostingEnvironment | IHostingEnvironment used to get the report stream from application **wwwroot\Resources** folder.

7. You cannot load the application report with path information in ASP.NET Core service. So, you should load the report as stream in **OnInitReportOptions**.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-
detail.rdl", System.IO FileMode.Open, System.IO FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}
```

8. You can replace the template code with the following code.

```
'csharp
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
    // Report viewer requires a memory cache to store the information of consecutive client request and
    // have the rendered report viewer information in server.
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
```

```
{  
    _cache = memoryCache;  
    _hostingEnvironment = hostingEnvironment;  
}  
  
// Post action to process the report from server based json parameters and send the result back to the  
client.  
  
[HttpPost]  
public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)  
{  
    return ReportHelper.ProcessReport(jsonArray, this, this._cache);  
}  
  
// Method will be called to initialize the report information to load the report with ReportHelper for  
processing.  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    string basePath = _hostingEnvironment.WebRootPath;  
  
    // Here, we have loaded the sales-order-detail.rdl report from application the folder  
    wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.  
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-  
    detail.rdl", System.IO FileMode.Open, System.IO FileAccess.Read);  
    reportOption.ReportModel.Stream = reportStream;  
}  
  
// Method will be called when reported is loaded with internally to start to layout process with  
ReportHelper.  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
}  
  
//Get action for getting resources from the report  
[ActionName("GetResource")]  
[AcceptVerbs("GET")]  
  
// Method will be called from Report Viewer client to get the image src for Image report item.  
public object GetResource(ReportResource resource)  
{  
    return ReportHelper.GetResource(resource, this, _cache);  
}
```

```
[HttpPost]  
public object PostFormReportAction()  
{  
    return ReportHelper.ProcessReport(null, this, _cache);  
}  
}  
`
```

The sales-order-detail.rdl report can be downloaded from [here](#). Also, you can add the report from Bold Reports installation location. For more information on installed sample location, see [Samples and demos](#).

9. Run the application and use the API URL in the Report Viewer reportServiceUrl property.

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

Enable Cross-Origin requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Call AddCors in Startup.ConfigureServices to add the CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp  
public void ConfigureServices(IServiceCollection services)  
{  
    services.AddMvc();  
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>  
    {  
        builder.AllowAnyOrigin()  
            .AllowAnyMethod()  
            .AllowAnyHeader();  
    });  
}
```

To specify the CORS policy for the controller, add the [EnableCors] attribute to the controller class. Specify the policy name.

```
'csharp
```

```
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]

public class ReportViewerController : Controller, IReportController
{
    public IActionResult Index()
    {
        return View();
    }

    ....
}
```

How to section for Angular Report Viewer

This section helps to get the answer for the frequently asked questions in Report Viewer.

- [Getting Started for earlier version](#)
- [How to create a RDL report](#)
- [How to create a RDLC report](#)
- [Change the exporting document file name based on the parameter](#)
- [Change the connection string datasource dynamically](#)
- [Disable the vertical scrollbar in parameter panel](#)

Display SSRS RDL report in Bold Reports Angular Report Viewer

This section explains you the steps required to create your first Angular reporting application in Angular CLI to display already created SSRS RDL report in Bold Reports Angular Report Viewer without using a Report Server, refer to the following steps.

If you are using higher version of [Bold Reports Angular](#) (>= v2.2.28), then refer [Getting Started for latest version](#).

Prerequisites

Before you begin, make sure your development environment includes the following:

- [Node JS](#) (version 8.x or 10.x)
- [NPM](#) (v3.x.x or higher)

Install the Angular CLI

Angular provides the easiest way to set Angular CLI projects using the [Angular CLI](#) tool. To install the CLI application globally to your machine, run the following command in the Command Prompt.

```
'typescript  
npm install -g @angular/cli  
'
```

To learn more about angular-cli commands, click [here](#).

Create a new application

To create a new Angular application, run the following command in the Command Prompt.

```
'typescript
```

```
ng new project-name
```

```
E.g : ng new reportviewerapp
```

```
'
```

The `ng new` command prompts you for information about features to include in the initial app project. Accept the defaults by pressing the Enter or Return key.

![Accept the initial app project defaults by pressing the Enter or Return key](/static/assets/angular/report-viewer/images/getting-started/angular-initial-app-features.png)

Configure Bold Report Viewer in Angular CLI

Bold reporting tools packages are distributed in NPM package as [@boldreports/angular-reporting-components](#).

1. To configure the Report Viewer component, change the directory to your application's root folder.

```
'typescript
```

```
cd project-name
```

```
E.g : cd reportviewerapp
```

```
'
```

2. Run the following commands to install the dependencies.

```
'typescript
```

```
npm install @boldreports/angular-reporting-components --save
```

```
'
```

3. Install the typings dependencies `jquery` and `boldreports/types`.

```
'typescript
```

```
npm install --save-dev @types/jquery
```

```
npm install --save-dev @boldreports/types
```

```
'
```

4. Add the typings `jquery`, and `reports.all` to the `src/tsconfig.app.json` file.

```
'js
```

```
{
...
...
"compilerOptions": {
...
...
"baseUrl": "",
"types": [
"jquery",
"reports.all"
]
},
...
...
}
`
```

5. Register the `@bold-reports/types` under the `typeRoots` node in the `tsconfig.json` file.

![Css script reference](/static/assets/angular/report-designer/images/register-typings.png)

If you are using Angular project version 8.1.x or higher, You should set the `enableIvy` option to `false` in the project's `tsconfig.json` file to resolve the Bold Reports package compiling issue.

6. Report Viewer requires `window.jQuery` object to render the component. Import jQuery in the `src/polyfills.ts` file as shown in the following code snippet.

```
`js
import * as jquery from 'jquery';
window['jQuery'] = jquery;
window['$'] = jquery;
`
```

Refer to the already configured import codes in the [webpack angular seed](#) application.

Adding CSS reference

Add Report Viewer component style (`bold.reports.all.min.css`) as given in the `angular.json` file within the `projectname -> styles` section (Eg. `reportviewerapp -> styles`).

If you are using Angular 6 or lower version project, add the changes in the `angular-cli.json` file.

```

`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "project": {
    "name": "reportviewerapp"
  },
  "reportviewerapp": [
    {
      "root": "src",
      "outDir": "dist",
      ...
      ...
      "styles": [
        "styles.css",
        "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css"
      ],
      "scripts": []
      ...
      ...
    }
  ]
}
```

```

In the previous code, the `material` theme is used. You can modify the theme based on your application, refer the following syntax: `./node_modules/@boldreports/javascript-reporting-controls/Content/[theme-name]/bold.reports.all.min.css`

### [Adding Report Viewer component](#)

To add the Report Viewer component, refer to the following steps:

1. Open the `app.module.ts` file.
2. You can replace the following code snippet in the `app.module.ts` file.

```

`typescript
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { BOLDREPORTVIEWERCOMPONENTS } from '@boldreports/angular-reporting-components/src/reportviewer.component';
```

```

```

import { AppComponent } from './app.component';
// data-visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/common/ej2-base.min.js';
import '@boldreports/javascript-reporting-controls/Scripts/common/ej2-data.min.js';
import '@boldreports/javascript-reporting-controls/Scripts/common/ej2-pdf-export.min.js';
import '@boldreports/javascript-reporting-controls/Scripts/common/ej2-svg-base.min.js';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min.js';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min.js';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
@NgModule({
declarations: [
AppComponent,
BOLDREPORTVIEWERCOMPONENTS
],
imports: [
BrowserModule
],
providers: [],
bootstrap: [AppComponent]
})
export class AppModule { }
`
```

3. Open the `app.component.html` file and initialize the Report Viewer.
4. You can replace the following code snippet in the `app.component.html` file.

```

`javascript
<bold-reportviewer id="reportViewer_Control" style="width: 100%;height: 950px">
</bold-reportviewer>
`
```

5. Open the `app.component.ts` and replace the following code example.

```
`typescript
```

```

import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'reportviewerapp';
  public serviceUrl: string;
  public reportPath: string;
  constructor() {
    // Initialize the Report Viewer properties here.
  }
}
`
```

If you have faced the issue 'ej' is not defined after the above configuration in Angular CLI latest version 7, refer to the following code snippet in your application where you have rendered Syncfusion Components(model file) to resolve the issue.

```

`typescript
/// <reference types="reports.all" />
`
```

Create Web API service

The Report Viewer requires a Web API service to process the report files. You can skip this step and use the online [Web API services](#) to preview the already available reports or you should create any one of the following Web API services:

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

Adding already created report

If you have created a new service, you can add the reports from the Bold Reports installation location. For more information, refer to [samples and demos](#) section.

1. Create a folder **Resources** in your Web API application to store the RDL reports and add already created reports to it.
2. Add already created reports to the newly created folder.

In this tutorial, the **sales-order-detail.rdl** report is used, and it can be downloaded in this [link](#).

Refer to the [create RDL report](#) section for creating new reports.

Set report path and Web API service

To set report path and Web API service, open the `app.component.ts` file and add the codes as shown in the constructor.

```
'typescript
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'jsreport-sample';
  public serviceUrl: string;
  public reportPath: string;
  constructor() {
    this.serviceUrl = 'https://demos.boldreports.com/services/api/ReportViewer';
    this.reportPath = '~/Resources/docs/sales-order-detail.rdl';
  }
}
`
```

In the above code, the `sales-order-detail.rdl` report and `reportServiceUrl` used from online URL.

Open the `app.component.html` to set `reportPath` and `reportServiceUrl` properties of Report Viewer as in the following.

```
'javascript
<bold-reportviewer id="reportViewer_Control" [reportServiceUrl] = "serviceUrl" [reportPath] =
"reportPath" style="width: 100%;height: 950px">
</bold-reportviewer>
`
```

Serve the application

To serve the application, follow these steps:

1. Navigate to the root of the application and run the application using the following command.

```
'typescript
```

ng serve

\

2. Navigate to the appropriate port `http://localhost:4200` in the browser.

See Also

[Render Report Server reports](#)

[Create RDLC report](#)

[Render RDLC reports](#)

[Preview report in print mode](#)

[Set data source credential for shared data sources](#)

[Change data source connection string](#)

[Production deployment](#)

Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

- Bold Reports Report Designer.
- Microsoft Report Builder.
- Visual Studio Report Server project template.

Bold Reports Report Designer

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

Microsoft SQL Report Builder

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

Visual Studio Report Server template

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

Create a RDLC report using business object data source

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

Prerequisites

- Microsoft Visual Studio 2017 or higher
- [Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

Create business object class

1. Open Visual Studio from the File menu and select **New Project**.
2. Create project with class library type from the project type list.

![Add a new class library project](/static/assets/angular/report-viewer/images/how-to/create-report/class-library-project.png)

3. Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
    public string ProdCat { get; set; }
    public string SubCat { get; set; }
    public string OrderYear { get; set; }
    public string OrderQtr { get; set; }
    public double Sales { get; set; }
}
```

4. Clean and build the application.

Add an RDLC report

1. Right-click the project and click **Add > New Item**.
2. Search Report with new item and select **Report Wizard** to start the report creation with dataset selection.
3. Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/angular/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

Data source and table configuration wizard

1. Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/angular/report-viewer/images/how-to/create-report/choose-data-source-type.png)

2. Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/angular/report-viewer/images/how-to/create-report/select-data-objects.png)

3. In the DataSet Properties wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/angular/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

4. Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/angular/report-viewer/images/how-to/create-report/arrange-table-fields.png)

5. Choose the table layout and click **Next**.
6. Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/angular/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/angular/report-viewer/images/how-to/create-report/sales-report-design.png)

To render the RDLC using Report Viewer, refer to the [RDLC Report](#) section.

How to change the exporting document file name based on parameter

Find the following steps to change the file based on parameter values in Report.

1. Create the file exporting file name using the parameters in **onRenderingComplete** event and store it in local variable.

```
'html
function onRenderingComplete(event) {
    var parameters = event.reportParameters;
    if(parameters){
        for (var i = 0; i < parameters.length; i++) {
```

```
if(parameters[i].Name == "Department"){
    this.exportFileName = "Sales for " + parameters[i].Value;
}
}
`
```

2. Use the file Name property with export, click event to change the file using the value stored in local variable used for having the file using the parameters.

```
`html
function onExportItemClick(event) {
    event.fileName = this.exportFileName ;
}
`
```

How to change the connection string datasource dynamically

Find the following steps to change the connection string in datasource.

1. You need a report action information to get the data source information from report. So, stored the JsonResult information to local property in **PostReportAction** method as shown in the following code example.

```
`csharp
private Dictionary<string, object> _jsonResult;
//Post action for processing the rdl/rdlc report
public object PostReportAction(Dictionary < string, object > jsonResult)
{
    if (jsonResult != null && jsonResult.Keys.Count > 0)
    {
        _jsonResult = jsonResult;
    }
    return ReportHelper.ProcessReport(jsonResult, this);
}
`
```

2. Use the JsonResult information with **ReportHelper.GetDatasource** API to get the data source details of the reports and you have to use **DataSourceCredentials** to change the connection string of the data source.

```
`csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (jsonResult != null && jsonResult.Keys.Count > 0)
{
List<DataSourceInfo> datasources = ReportHelper.GetDataSources(_jsonResult, this);
foreach (DataSourceInfo item in datasources)
{
if (item.DataProvider == "SQL")
{
string connectionString = "Data Source = dataplatformdemodata.syncfusion.com; Initial Catalog =
AdventureWorks; User ID = 'demoreadonly@data-platform-demo'; Password = 'N@c)=Y8s*1&dh'";
DataSourceCredentials DataSourceCredentials = new DataSourceCredentials();
DataSourceCredentials.Name = item.DataSourceName;
DataSourceCredentials.UserId = null;
DataSourceCredentials.Password = null;
DataSourceCredentials.ConnectionString = connectionString;
DataSourceCredentials.IntegratedSecurity = false;//if windows credentials means we need to pass true
as IntegratedSecurity
reportOption.ReportModel.DataSourceCredentials = new List<DataSourceCredentials>
{
DataSourceCredentials
};
}
}
}
}`
```

How to disable the vertical scrollbar in parameter panel

To disable the vertical scrollbar in parameter panel, set the `enableparameterblockscroller` property to false.

Example

'js

<div id="report viewer"></div>

```
<script>
  $("#report viewer").boldReportViewer(
  {
    enableParameterBlockScroller: false
  });
</script>
```

Reporting tools for Angular

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

Key features

Data Sources --- Supports connection to major data providers such as

Microsoft SQL Server, Oracle, OLEDB and **ODBC** for exploring data and design reports with a wide range of data sources.

User friendly Environment --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

Report items --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

Report Parameter --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

Expression --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

Add Web Report Designer to an Angular application using Angular CLI

This section explains the steps required to add Web Report Designer to an Angular application.

If you are using lower version of [Bold Reports Angular](#) (< v2.2.28), then refer [Getting Started for earlier version](#).

Prerequisites

Before getting started with Bold Report Designer, make sure your development environment includes the following,

- [Node JS](#) (version 8.x or 10.x)
- [NPM](#) (v3.x.x or higher)

Install the Angular CLI

Angular provides the easiest way to set Angular CLI projects using [Angular CLI](#) tool. To install the CLI application globally to your machine, run the following command in Command Prompt.

```
'typescript
npm install -g @angular/cli@latest
'
```

To learn more about angular-cli commands refer [here](#)

Create a new application

To create a new Angular application run the below command in Command Prompt.

```
'typescript
ng new project-name
E.g : ng new reportdesignerapp
'
```

The `ng new` command prompts you for information about features to include in the initial app project. Add the Angular Routing to your angular application by entering `y` in the prompt window then press **Enter** key. Now, choose the CSS stylesheet format using the arrow keys and then press **Enter** key. The Angular CLI installs the required Angular npm packages and other dependencies.

Configure Bold Report Designer in Angular CLI

Bold Reporting packages are distributed in npm as [@boldreports/angular-reporting-components](#).

1. To configure the web Report Designer component, change the directory to your application's root folder.

```
'typescript
cd project-name
E.g : cd reportdesignerapp
'
```

2. Run the following commands to install the [Bold Reports Angular](#) library.

```
'typescript
npm install @boldreports/angular-reporting-components --save-dev
'
```

3. Also, Install [Bold Reports typings](#) by executing the below command.

```
'typescript
npm install --save-dev @boldreports/types
'
```

4. Register the `@bold-reports/types` under the `typeRoots` and add the `typings jquery` and `reports.all` to the `tsconfig.app.json` file.

```
`js
{
...
...
"compilerOptions": {
...
...
"typeRoots": [
"node_modules/@types",
"node_modules/@boldreports/types"
],
"types": [
"jquery",
"reports.all"
]
},
...
...
}`
```

5. Report Designer requires `window.jQuery` object to render the component. Import jQuery in `src/polyfills.ts` file as in the following code snippet.

```
`js
import * as jquery from 'jquery';
window['jQuery'] = jquery;
window['$'] = jquery;
`
```

Adding CSS reference

1. Open the `angular.json` file from your application's root directory.
2. Refer the web Report Designer component style scripts `bold.reports.all.min.css` and `bold.reportdesigner.min.css` under the `styles` node of `projects` section as in the below code example.

If you are using Angular 6 or lower version project, add the changes in `angular-cli.json` file.

```
![Css script reference](/static/assets/angular/report-designer/images/Getting-Started_img1.png)
`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "reportdesignerapp": {
      "root": "",
      "sourceRoot": "src",
      "projectType": "application",
      "prefix": "app",
      "schematics": {},
      "architect": {
        "build": {
          "builder": "@angular-devkit/build-angular:browser",
          "options": {
            "outputPath": "dist/reportdesignerapp",
            "index": "src/index.html",
            "main": "src/main.ts",
            "polyfills": "src/polyfills.ts",
            "tsConfig": "src/tsconfig.app.json",
            "assets": [
              "src/favicon.ico",
              "src/assets"
            ],
            "styles": [

```

```

    "src/styles.css",
    "./node_modules/@boldreports/javascript-reporting-
    controls/Content/material/bold.reports.all.min.css",
    "./node_modules/@boldreports/javascript-reporting-
    controls/Content/material/bold.reportdesigner.min.css"
],
"scripts": [],
"es5BrowserSupport": true
},
...
...
}
`
```

In the above code snippet the `material` theme is used. You can modify the theme based on your application, refer the following syntax: `./node_modules/@boldreports/javascript-reporting-controls/Content/[theme-name]/bold.reports.all.min.css` and `./node_modules/@boldreports/javascript-reporting-controls/Content/[theme-name]/bold.reportdesigner.min.css`

[Adding code mirror reference](#)

Report Designer requires the code mirror scripts and styles to edit the SQL queries and Visual Basic code functions with syntax highlighter.

1. Run the below command, to install the code mirror dependencies.

```

`typescript
npm install codemirror
`
```

2. Open the `angular.json` file from your application's root directory and Refer the code mirror script and styles under the `scripts` and `styles` node of `projects` section as in the below code example.

```

![Css script reference](/static/assets/angular/report-designer/images/code-mirror-references.png)
`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
```

```
"projects": {  
  "reportdesignerapp": {  
    "root": "",  
    "sourceRoot": "src",  
    "projectType": "application",  
    "prefix": "app",  
    "schematics": {},  
    "architect": {  
      "build": {  
        "builder": "@angular-devkit/build-angular:browser",  
        "options": {  
          "outputPath": "dist/reportdesignerapp",  
          "index": "src/index.html",  
          "main": "src/main.ts",  
          "polyfills": "src/polyfills.ts",  
          "tsConfig": "src/tsconfig.app.json",  
          "assets": [  
            "src/favicon.ico",  
            "src/assets"  
          ],  
          "styles": [  
            "src/styles.css",  
            "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css",  
            "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css",  
            "./node_modules/codemirror/lib/codemirror.css",  
            "./node_modules/codemirror/addon/hint/show-hint.css"  
          ],  
          "scripts": [  
            "./node_modules/codemirror/lib/codemirror.js",  
            "./node_modules/codemirror/addon/hint/show-hint.js",  
            "./node_modules/codemirror/addon/hint/sql-hint.js",  
            "./node_modules/codemirror/mode/vb/vb.js"  
          ]  
        }  
      }  
    }  
  }  
}
```

```
],
"es5BrowserSupport": true
},
...
...
}
`
```

Adding Scripts reference

Refer the following scripts in your angular application. To render chart in Report Designer, we need to add ej.chart.min.js additionally. To render gauge in Report Designer, we need to add the following scripts ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js.

```
'basic
bold.reports.common.min.js
bold.reports.widgets.min.js
bold.report-designer-widgets.min.js
bold.report-viewer.min.js
bold.report-designer.min.js
`
```

1. Open the angular.json file from your application's root directory.
2. Refer above mentioned list of scripts under the scripts node of projects section as in the below code example.

If you are using Angular 6 or lower version project, add the changes in angular-cli.json file.

![Css script reference](/static/assets/angular/report-designer/images/script-reference.png)

```
`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "reportdesignerapp": {
      "root": "",
      "sourceRoot": "src",
      "projectType": "application",
```

```
"prefix": "app",
"schematics": {},
"architect": {
  "build": {
    "builder": "@angular-devkit/build-angular:browser",
    "options": {
      "outputPath": "dist/reportdesignerapp",
      "index": "src/index.html",
      "main": "src/main.ts",
      "polyfills": "src/polyfills.ts",
      "tsConfig": "src/tsconfig.app.json",
      "assets": [
        "src/favicon.ico",
        "src/assets"
      ],
      "styles": [
        "src/styles.css",
        "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css",
        "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css",
        "./node_modules/codemirror/lib/codemirror.css",
        "./node_modules/codemirror/addon/hint/show-hint.css"
      ],
      "scripts": [
        "./node_modules/codemirror/lib/codemirror.js",
        "./node_modules/codemirror/addon/hint/show-hint.js",
        "./node_modules/codemirror/addon/hint/sql-hint.js",
        "./node_modules/codemirror/mode/vb/vb.js",
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/ej2-base.min.js",
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/ej2-data.min.js",
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/ej2-pdf-export.min.js",
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/ej2-svg-base.min.js",
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min.js"
      ]
    }
  }
}
```

```
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min.js",
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/bold.reports.common.min.js",
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/bold.reports.widgets.min.js",
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/bold.report-designer-widgets.min.js",
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min.js",
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min.js",
"./node_modules/@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min.js"
],
"es5BrowserSupport": true
},
...
...
}
`
```

Adding Report Designer component

To add the web Report Designer component refer the following steps:

1. Open the `src/app/app.module.ts` file.
2. Import `BoldReportDesignerModule` from `@boldreports/angular-reporting-components` package as in the below code snippet.

```
'typescript
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { BoldReportDesignerModule } from '@boldreports/angular-reporting-components';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
```

```
BrowserModule,  
BoldReportDesignerModule  
],  
providers: [],  
bootstrap: [AppComponent]  
}  
  
export class AppModule { }  
'
```

3. Open the `src/app/app.component.html` file and initialize the web Report Designer.
4. You can replace the following code snippet in the `app.component.html` file.

```
'html  
<bold-reportdesigner id="designer" style="position: absolute; height: 550px; width: 1250px;">  
</bold-reportdesigner>  
'
```

5. Open the `src/app/app.component.ts` and replace the below code example.

```
'typescript  
import { Component } from '@angular/core';  
@Component({  
  selector: 'app-root',  
  templateUrl: './app.component.html',  
  styleUrls: ['./app.component.css']  
})  
export class AppComponent {  
  title = 'reportdesignerapp';  
  public serviceUrl: string;  
  constructor() {  
    // Initialize the Report Designer properties here.  
  }  
}
```

If you have faced the issue 'ej' is not defined after the above configuration in Angular CLI latest version 7, Please refer the below code snippet in your application where you have rendered Bold Reporting Components(model file) to resolve the issue.

```
'typescript
```

```
/// <reference types="reports.all" />
`
```

Create Web API service

The web Report Designer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API service.

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

Set Web API service URL

To set Web API service, open the app.component.ts file and add the code snippet as in the constructor.

```
'typescript
```

```
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'reportdesignerapp';
  public serviceUrl: string;
  constructor() {
    this.serviceUrl = "https://demos.boldreports.com/services/api/ReportingAPI";
  }
}
```

Open the app.component.html to set ServiceUrl property of web Report Designer as in the following code snippet.

```
'javascript
```

```
<bold-reportdesigner id="designer" [serviceUrl] = "serviceUrl" style="position: absolute; height: 550px; width: 1250px;">
```

Localization

Serve the application

```
</bold-reportdesigner>
```

Serve the application

1. Go to the workspace folder (reportdesignerapp).
2. Launch the server by using the CLI command **ng serve**, with the --open option.

```
'typescript
```

```
ng serve --open
```

3. The **ng serve** command launches the server and the **--open** option automatically opens your browser to <http://localhost:4200/>

![report designer output](/static/assets/angular/report-designer/images/Getting-Started_img9.png)

See Also

[Production deployment](#)

Localization

Localization is the process of customizing the application for a given culture and locale - involving much more than the simple translation of text. In web report designer, localized strings appropriate to each culture are stored in separate files and loaded according to the UI culture setting.

By default report designer display strings in US English(en-US).

To render the static text with specific culture, refer the following corresponding culture script files and set culture name to the **locale** property of the Report Designer.

- ej.localetexts.fr-FR.min.js
- ej.culture.fr-FR.min.js
- Run the below command, to install the **@boldreports/global** package

```
'typescript
```

```
npm install @boldreports/global --save
```

- Refer the **ej.localetexts.fr-FR.min.js** and **ej.culture.fr-FR.min.js** from **node_modules** in **app.module.ts** file .

```
'ts
```

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { BoldReportsModule } from '@boldreports/angular-reporting-components';
```

```
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
import '@boldreports/global/I10n/reportdesigner/ej.localetexts.fr-FR.min.js';
import '@boldreports/global/i18n/ej.culture.fr-FR.min.js';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    BoldReportsModule,
    AppRoutingModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

If we import the culture before the `@boldreports/angular-reporting-components` package, we get the following error in our application. So, we should import the culture after the `@boldreports/angular-reporting-components` package.

![Angular culture error](/static/assets/angular/report-designer/images/error/cultureerror.png)

- Open the `src/app/app.component.ts` and initialize the `locale` for Report Designer.

```
'ts
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
```

```
public serviceUrl: string;  
public locale: string;  
constructor() {  
    this.serviceUrl = "https://demos.boldreports.com/services/api/ReportingAPI";  
    this.locale = "fr-FR";  
}  
}  
`
```

- Open the `src/app/app.component.html` file and assign the `locale` for Report Designer.

```
`javascript  
<bold-reportdesigner id="designer" style="position: absolute; height: 550px; width: 1250px;" [serviceUrl]  
= "serviceUrl" [locale]= "locale"></bold-reportdesigner>  
`
```

Integrate the component with Report Server

You can integrate Report Designer with Report Server to create, edit, browse and publish reports using the Report Server built-in API service.

If you don't have Report Designer application then create the app using the [Getting-started](#) steps.

- The Report Designer requires the `serviceAuthorizationToken` to perform the API actions with Bold Report Server. Create a token for the user by using the server RESTful API, the following code is used to generate the token.
 1. Run the below command to install the `HTTP` dependencies.

```
`typescript  
npm install @angular/http@latest  
`
```

2. Open the `app.module.ts` file and import `HTTP` module.

```
`typescript  
...  
import { HttpClientModule } from '@angular/http';  
...  
...  
imports: [  
    BrowserModule,
```

```
HttpModule  
],  
providers: [],  
bootstrap: [AppComponent]  
})  
...  
,
```

3. Open the `app.component.ts` and create the user token using `http.post`. You can replace the following code example in your application.

```
'js  
import { Component } from '@angular/core';  
import { Http, Response, Headers } from '@angular/http';  
const httpOptions = {  
headers: new Headers({  
'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8'  
})  
};  
class ReportserviceService {  
private http: Http;  
constructor(http: Http) {  
this.http = http;  
}  
async getAsyncToken(serverUrl: string, userName: string, password: string): Promise<string>{  
return await this.http.post(serverUrl + '/get-user-key', {userid : userName,password : password  
}).toPromise().then(this.extractData);  
}  
private extractData(res: Response): string {  
var token = JSON.parse(res.json().Token);  
return token["tokentype"] + " " + token["accesstoken"];  
}  
}  
@Component({  
selector: 'app-root',
```

```
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
public serviceUrl: string;
public serverServiceAuthorizationToken: string;
public serverURL: string;
private tokenService: ReportserviceService
constructor(private http: Http) {
this.tokenService = new ReportserviceService(this.http);
let self = this
this.serverURL = 'https://on-premise-demo.boldreports.com/reporting/api/site/site1';
this.serviceUrl ='https://on-premise-demo.boldreports.com/reporting/reportservice/api/Designer';
this.tokenService.getAsyncToken(this.serverURL, "guest@boldreports.com", "demo").then(function
(data) {
self.serverServiceAuthorizationToken = data;
});
}
onAjaxRequest(event) {
event.headers.push({ Key: 'serverurl', Value: this.serverURL });
}
}
`
```

4. Open the `app.component.html` and set the Bold Report Server built-in service URL to the `serviceUrl` property and assign the created token to `serviceAuthorizationToken` properties as in the below code example.

```
`js
<bold-reportdesigner
id="reportDesigner_Control"
[serviceUrl]="serviceUrl"
[serviceAuthorizationToken] = "serverServiceAuthorizationToken"
(ajaxBeforeLoad) = "onAjaxRequest($event)" style="position: absolute;height: 550px; width: 1250px;">
</bold-reportdesigner>
```

5. Navigate to the root of the application and run the application using the following command.

```
'typescript
```

```
ng serve
```

Designing Reports

The Bold Report Designer provides an end-user documentation that describes how to interact with the Report Designer's UI and design an interactive reports. Refer the following user guide sections to get start with Bold Report Designer.

Connecting your data

A report must have a data source and dataset to include data. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

[Manage data source](#)

[Manage data set](#)

Transform your data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)

[Formatting columns](#)

[Query filter](#)

[Data set parameter](#)

[Query parameter](#)

[Configure expression columns](#)

Working with report parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it.

[Add report parameter](#)

[Edit report parameter](#)

[Delete report parameter](#)

[Cascading parameter](#)

[Multi-value parameter](#)

Embed images into the report

Supports to embed local or database images into the report and image data is stored within the report definition. The embedded images in a report are listed in the **Image Manager panel**.

[Embed images into the report](#)

Interacting with report design surface

WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

[Interacting with report design surface](#)

Report items

Offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

[Configure report items](#)

Customize appearance

The Properties panel shows all the properties of the selected section, report items or the report itself to customize the appearance of the report.

[Customize appearance](#)

Report design settings

[Configure design settings](#)

Shape report data

[Filter data](#)

[Group data](#)

[Sort data](#)

[Format data](#)

Interactive features

[Hyperlink](#)

[Drilldown](#)

[Interactive sorting](#)

Working with Expressions

Expressions are used to specify criteria for retrieving and formatting data, creating calculated fields and calculating summaries, conditionally shaping data and changing a report control's appearance.

[Expressions builder](#)

Open report

[Open report](#)

Save report

[Save report](#)

Preview report

[Preview report](#)

Migrate Report Designer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Designer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Designer from Syncfusion Essential Studio release version to Bold Reports version of Angular Report Designer application.

NPM packages

| Platform | Old Package | New Package |
|------------|-----------------------|------------------------------------------------------------|
| Javascript | syncfusion-javascript | @boldreports/javascript-reporting-controls |
| Angular | ej-angular2 | @boldreports/angular-reporting-components |
| Globalize | syncfusion-ej-global | @boldreports/global |

The above Javascript NPM packages contain the following scripts and styles required for Reporting components.

| Reporting Component | Styles | Scripts | Usage |
|---------------------|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Report Viewer | bold.reports.all.min.css | bold.reports.common.min.js bold.reports.widgets.min.js ej.chart.min.js ej2-base.min.js ej2-data.min.js ej2-pdf-export.min.js ej2-svg-base.min.js ej2-lineargauge.min.js ej2-circulargauge.min.js ej.map.min.js bold.report-viewer.min.js | bold.reports.all.min.css - It contains the styles and css references of reporting dependent components. bold.reports.common.min.js - Common script for reporting widgets. bold.reports.widgets.min.js - It contains the scripts of dependent Syncfusion controls that are common for both Report Designer and Report Viewer. ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart report item. ej2-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item. ej2-data.min.js - Renders the gauge item. Add this script only if your report contains |

| | | | |
|-----------------|-----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p>the gauge report item.</p> <p><code>ej2-pdf-export.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-svg-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-lineargauge.min.js</code> - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p><code>ej2-circulargauge.min.js</code> - Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Renders the Syncfusion Report Viewer widget.</p> |
| Report Designer | <code>bold.reports.all.min.css</code> <code>bold.reportdesigner.min.css</code> | <code>bold.reports.common.min.js</code> <code>bold.reports.widgets.min.js</code> <code>bold.report-designer-widgets.min.js</code> <code>ej.chart.min.js</code> <code>ej2-base.min.js</code> | <code>bold.reports.all.min.css</code> - It contains the styles and css references of reporting dependent components. <code>bold.reportdesigner.min.css</code> - It contains the styles and css references of Report Designer Component. <code>bold.reports.common.min.js</code> |

| | | | |
|--|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | <p>ej2-data.min.js</p> <p>ej2-pdf-export.min.js</p> <p>ej2-svg-base.min.js</p> <p>ej2-lineargauge.min.js</p> <p>ej2-circulargauge.min.js</p> <p>ej.map.min.js</p> <p>bold.report-viewer.min.js</p> <p>bold.report-designer.min.js</p> | <p>.js - Common script for reporting widgets.</p> <p>bold.reports.widgets.min.js - It contains the scripts of dependent controls that are common for both Report Designer and Report Viewer.</p> <p>bold.report-designer-widgets.min.js - It contains the scripts of Report Designer dependent controls.</p> <p>ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart report item.</p> <p>ej2-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-data.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-pdf-export.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-svg-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-lineargauge.min.js - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p>ej2-</p> |
|--|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| | | | |
|--|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | | <p><code>circulargauge.min.js</code> - Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Previews the reports designed with Report Designer.</p> <p><code>bold.report-designer.min.js</code> - Renders the Syncfusion Report Designer widget.</p> |
|--|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Server-side migration

Assemblies

| Old Assemblies | New Assemblies | Description |
|----------------------------------------------------------------------------------------------|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>Syncfusion.EJ.ReportViewer.dll</code> <code>Syncfusion.EJ.ReportDesigner.dll</code> | <code>BoldReports.Web.dll</code> | The <code>Syncfusion.EJ.ReportViewer.dll</code> and <code>Syncfusion.EJ.ReportDesigner.dll</code> assemblies have been combined as <code>BoldReports.Web.dll</code> . |

Packages

| Old Packages | New Packages |
|--------------------------------------------|------------------------------|
| <code>Syncfusion.Web.ReportDesigner</code> | |
| <code>Syncfusion.Web.ReportViewer</code> | <code>BoldReports.Web</code> |

Namespace changes

| Assembly Name | Old Namespace | New Namespace |
|----------------------------------|-----------------------------------------|-------------------------------------------|
| <code>BoldReports.Web.dll</code> | <code>Syncfusion.Reports.EJ</code> | <code>BoldReports.Web</code> |
| | <code>Syncfusion.EJ.ReportWriter</code> | <code>BoldReports.Writer</code> |
| | <code>Syncfusion.EJ.ReportViewer</code> | <code>BoldReports.Web.ReportViewer</code> |

| | | |
|--|-----------------------------------|--------------------------------|
| | Syncfusion.EJ.ReportDesigner | BoldReports.Web.ReportDesigner |
| | Syncfusion.Reports.EJ.Data | BoldReports.Data |
| | Syncfusion.Reporting | BoldReports.Configuration |
| | Syncfusion.EJ.RDL.ServerProcessor | BoldReports.ServerProcessor |

Based on above assembly and namespace changes, modify the Report Designer **Web API Controller** in your application.

Control initialization

| Old code snippet | New code snippet |
|-------------------------------------------------------|-----------------------------------------------------------|
| <ej-reportdesigner id="designer"></ej-reportdesigner> | <bold-reportdesigner id="designer"></bold-reportdesigner> |

Report export configuration

The **BoldReports.Web** assembly can export the reports with data visualization components such as chart, gauge, and map, only if we configure the web scripts in Report Designer Web API controller. To configure the scripts in Web API controller, refer to the following steps:

1. Open the Report Designer Web API controller.
2. Configure the following scripts and styles in **OnInitReportOptions** on Web API controller:
 - o jquery-1.7.1.min.js
 - o bold.reports.common.min.js
 - o bold.reports.widgets.min.js
 - o ej.chart.min.js - Exports the chart item. Add this script only if your report contains the chart report item.
 - o ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js - Exports the gauge item. Add this script only if your report contains the gauge report item.
 - o ej.map.min.js - Exports the map item. Add this script only if your report contains the map report item.
 - o bold.report-viewer.min.js
3. You can replace the **OnInitReportOptions** action in Report Designer Web API controller using below code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {

```

```
resourcesPath + @"\\bold-reports\\common\\bold.reports.common.min.js",
resourcesPath + @"\\bold-reports\\common\\bold.reports.widgets.min.js",
//Chart component script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.chart.min.js",
//Gauge component scripts
resourcesPath + @"\\bold-reports\\common\\ej2-base.min.js",
resourcesPath + @"\\bold-reports\\common\\ej2-data.min.js",
resourcesPath + @"\\bold-reports\\common\\ej2-pdf-export.min.js",
resourcesPath + @"\\bold-reports\\common\\ej2-svg-base.min.js",
resourcesPath + @"\\bold-reports\\data-visualization\\ej2-lineargauge.min.js",
resourcesPath + @"\\bold-reports\\data-visualization\\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.chart.min.js",
resourcesPath + @"\\bold-reports\\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\\jquery-1.7.1.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

NuGet Packages for Angular

Refer to the following steps to configure Bold Reporting tools NuGet packages for Angular Web API application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting tools NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

1. Open Visual Studio application.
2. On the **Tools** menu, select **Options**.
3. Expand the **NuGet Package Manager** and select **Package Sources**.

4. Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: NuGet.org

Source: https://api.nuget.org/v3/index.json

![Online NuGet Configure](/static/assets/angular/report-designer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reporting tools NuGet packages are shipped into our Report Platform SDK build. To configure the packages from Bold Reports installed location, use the following steps:

1. Open your Visual Studio application.
2. On the **Tools** menu, select **Options**.
3. Expand the **NuGet Package Manager**, and then select **Package Sources**.
4. Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages

![Offline NuGet Configure](/static/assets/angular/report-designer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

Installing NuGet packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

1. On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**. Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.
2. By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages **BoldReports.Web**, and then click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

1. On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.
2. Run the following NuGet installation commands:

```
'cmd
```

install specified package in default project

Upgrading NuGet packages

install specified package in default project

Install-Package <Package Name>

install specified package in default project with specified package source

Install-Package <Package Name> -Source <Source Location>

install specified package in specified project

Install-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

install specified package in default project

Install-Package BoldReports.Web

install specified package in default project with specified Package Source

Install-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

install specified package in specified project

Install-Package BoldReports.Web -ProjectName BoldReportingApplication

Upgrading NuGet packages

Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

1. On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....** Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.
2. Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

Upgrading using Package Manager Console

To update the installed Bold Reporting tools NuGet packages using the Package Manager Console:

1. On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.
2. Run the following NuGet installation commands:

`cmd

Update specific NuGet package in default project

Update-Package <Package Name>

Update all the packages in default project

Update-Package

Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

Update specified Bold Reporting NuGet package

Update-Package BoldReports.Web

Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportingApplication

Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

1. Download the latest [NuGet CLI](#).

To update the existing nuget.exe to latest version use the following command:

`cmd

nuget update -self

2. Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting tools NuGet packages.

update all NuGet packages from config file

Upgrading NuGet packages

`cmd

update all NuGet packages from config file

```
nuget update <configPath> [options]
```

update all NuGet packages from specified Packages Source

`nuget update -Source <Source Location> [optional]`

`configPath` is optional. It identifies the `package.config` or `solutions` file lists the packages utilized in the project.

For example:

`cmd

Update all NuGet packages from config file

```
nuget update "C:\Users\SyncfusionApplication\package.config"
```

Update all NuGet packages from specified Packages Source

```
nuget update -Source "https://api.nuget.org/v3/index.json"
```

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

Deployment

This topic explains about simplest production-ready deployment of your Angular Report Designer application to a remote server by creating a production build.

To create a production build run the following command and copy the output directory to a web server.

`typescript

```
node modules\.bin\ng build --prod
```

1

Please refer to the [angular deployment](#) documentation to know more about deploying your Angular application.

If you get error `FATAL ERROR: CALLANDRETRYLAST Allocation failed - JavaScript heap out of memory` add `--maxoldspacesize=xx` space in `ngc.cmd` and `ng.cmd` file from `node_modules\bin` folder.

Modify ngc.cmd

1

```
@IF EXIST "%~dp0\node.exe" (
```

"%~dp0\ngit\cmd\git.exe" --max-oldspace-size=8192 "%~dp0\.\\@angular\\compiler-cli\\src\\main.js" %*

```
) ELSE (
@SETLOCAL
@SET PATHEXT=%PATHEXT%;.JS;=%
node --maxoldspace_size=8192 "%~dp0..\@angular\compiler-cli\src\main.js" %*
)
`
```

Modify ng.cmd

```
`js
@if exist "%~dp0\node.exe" (
"%~dp0\node.exe" --maxoldspace_size=8192 "%~dp0..\@angular\cli\bin\ng" %*
) ELSE (
@SETLOCAL
@SET PATHEXT=%PATHEXT%;.JS;=%
node --maxoldspace_size=8192 "%~dp0..\@angular\cli\bin\ng" %*
)
`
```

Responsive layout rendering of Angular Report Designer

Report Designer will adaptively render itself with optimal user interfaces for tablet or desktop form factors. It has built-in responsive support, so that it will adjust automatically based on the browser viewport. Setting width to 100% is simply enough to make it responsive. This helps your application to scale elegantly on all form factors with ease.

Normal layout

The following output shows the normal layout rendering of Report Designer tool bar items.

![Rendering of Report Designer tool bar items in normal layout](/static/assets/angular/report-designer/images/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Designer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/angular/report-designer/images/responsive-layout.png)

Samples and demos

Browse and explore our ready-to-use the designer samples, online and offline demos.

Locally installed reports

You can obtain sample rdl and rdlc files from Bold Reports installed location

%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view the Angular Report Designer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view the GitHub Report Designer demo samples.

Supported Browsers

This document provides the list of web browsers supported by the Web Report Designer

| Desktop Browsers | Version |
|-------------------|---------|
| ----- | ----- |
| Internet Explorer | 11.0+ |
| Microsoft Edge | Current |
| Firefox | 22+ |
| Google Chrome | 17+ |
| Opera | 12+ |
| Safari | 5+ |

Angular Report Designer Reporting Service

The Angular Report Designer requires a Web API service to process the data and file actions. The following topics explains how to create reporting Web API service to create, edit, and browse reports.

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

IReportDesignerController

The interface `IReportDesignerController` has the declaration of action methods that are defined in Web API Controller to retrieve data, save, edit and browse reports from the server. The `IReportDesignerController` has the following action methods declaration.

| Methods | Description |
|--------------------|------------------------------------------------------------------------|
| PostDesignerAction | Action (HttpPost) method for posting the request for designer actions. |
| UploadReportAction | Action (HttpPost) method for posted file actions. |
| SetData | Write the resource into storage location. |
| GetData | Read the resource from storage location. |
| GetImage | Action (HttpGet) method for getting resource for report images. |
| PostReportAction | Action (HttpPost) method for posting the request for report process. |

| | | |
|---------------------|------------------------------------------------------------------------------------|--|
| GetResource | Action (HttpGet) method for getting resource for report. | |
| OnInitReportOptions | Report initialization method that is triggered when report begins to be processed. | |
| OnReportLoaded | Report loaded method that is triggered when report and sub report begin loading. | |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| Methods | Description | |
|---------------|------------------------------------------------------|--|
| ----- ----- | | |
| GetResource | Returns the report resource for the requested key. | |
| ProcessReport | Processes the report request and returns the result. | |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request from the Report Viewer control and returns the response to the Report Viewer control. It has the following methods.

Methods | Description

GetResource | Returns the report resource for the requested key.

ProcessReport | Processes the report request and returns the result.

`csharp

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using BoldReports.Web;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
namespace ReportDesignerAPIService
{
    [System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
    public class ReportingAPIController : ApiController, IReportDesignerController
    {
        /// <summary>
```

```
/// Action (HttpGet) method for getting resource for report images.  
/// </summary>  
/// <param name="key">The unique key for request identification.</param>  
/// <param name="image">The name of requested image.</param>  
/// <returns>The object data.</returns>  
[System.Web.Http.ActionName("GetImage")]  
[AcceptVerbs("GET")]  
public object GetImage(string key, string image)  
{  
    return ReportDesignerHelper.GetImage(key, image, this);  
}  
/// <summary>  
/// Action (HttpPost) method for posting the request for designer actions.  
/// </summary>  
/// <param name="jsonData">The JSON data posted for processing designer request.</param>  
/// <returns>The object data.</returns>  
public object PostDesignerAction(Dictionary<string, object> jsonData)  
{  
    //Processes the designer request and returns the result.  
    return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);  
}  
/// <summary>  
/// File upload method that is call while upload a file.  
/// </summary>  
/// <param name="key">The unique key for report designer</param>  
/// <param name="itemId">The unique key to get the required resource.</param>  
/// <param name="itemData">Gets the resource data.</param>  
/// <param name="errorMessage">Returns the error message, if the write action is failed.</param>  
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)  
{  
    //You can implement the file save logic  
}  
/// <summary>
```

```
/// File download method that is call while downloading a file.  
/// </summary>  
/// <param name="key">The unique key for report designer</param>  
/// <param name="itemId">The unique key to get the required resource.</param>  
public ResourceInfo GetData(string key, string itemId)  
{  
    //You can implement the file download logic  
}  
/// <summary>  
/// Action (HttpPost) method for uploaded file actions.  
/// </summary>  
public void UploadReportAction()  
{  
    //Processes the designer file upload request's.  
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);  
}  
/// <summary>  
/// Action (HttpGet) method for getting resource to report.  
/// </summary>  
/// <param name="key">The unique key to get the required resource.</param>  
/// <param name="resourcetype">The type of the requested resource.</param>  
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>  
/// <returns>The object data.</returns>  
[System.Web.Http.ActionName("GetResource")]  
[AcceptVerbs("GET")]  
public object GetResource(string key, string resourcetype, bool isPrint)  
{  
    //Returns the report resource for the requested key.  
    return ReportHelper.GetResource(key, resourcetype, isPrint);  
}  
/// <summary>  
/// Report Initialization method that is triggered when report begin processed.
```

```
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //You can update report options here
}

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
    //Processes the report request and returns the result.
    return ReportHelper.ProcessReport(jsonData, this as IReportController);
}
}
}
`
```

Create ASP.NET Web API Service

In this section, you will learn how to create a Web API Service for Report Designer using the new ASP.NET Empty Web Application template.

1. Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.
2. Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

3. Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![Creating a new ASP.NET Web Application Project](/static/assets/angular/report-designer/report-service/web-api-images/aspnet-new-project.png)

4. Then choose the empty application in template and click OK.

![Choosing a empty template in new Project](/static/assets/angular/report-designer/report-service/web-api-images/aspnet-empty-application.png)

List of dependency Libraries

The Web API service configuration requires reporting server-side assembly references.

1. Right-click the project/solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages...**

![Open nuget manager](/static/assets/angular/report-designer/report-service/web-api-images/add-dependencies-nuget.png)

Alternatively, click **Tools** in menu, then select **NuGet Package Manager | Manage NuGet Packages for Solution...**

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

2. Search the **BoldReports.Web** in the **Browse** tab and install **BoldReports.Web** NuGet package in the application.
3. **BoldReports.Web** package will install the following required dependencies for Report Designer into your application. Click OK.

Package | Purpose

BoldReports.Web | Builds the server-side implementations.

4. The following table provides details about the dependency packages and its usage.

Packages | Purpose

Syncfusion.Pdf.AspNet | Exports the report to a PDF.

Syncfusion.DocIO.AspNet | Exports the report to a Word.

Syncfusion.XlsIO.AspNet | Exports the report to an Excel.

Syncfusion.Presentation.AspNet | Exports the report to an PowerPoint.

5. Install the **Microsoft.AspNet.WebApi.Cors** from Nuget to enable the Cross-Origin Resource Sharing (CORS) for Web API.

```
'csharp
Install-Package Microsoft.AspNet.WebApi.Cors
'
```

This command installs the latest package and updates all dependencies, including the core Web API libraries. Use the -Version flag to target a specific version. Browser security prevents Report Designer making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Add Routing Information

The following steps guide you to configure the routing to include an action name in the URI.

1. Right-click the project in the solution explorer and select **Add > New item**.
2. In the Add New Item window, select **Global Application class** and name it as **Global.asax**, and then click Add.

![Adding Global.asax file](/static/assets/angular/report-designer/report-service/web-api-images/add-global-application-class.png)

3. Open the code-behind file **Global.asax.cs** and add the following using statement.

```
'csharp
using System.Web.Http;
'
```

4. Then add the the following code to the **Application_Start** method to register CORS.

```
'csharp
protected void Application_Start(object sender, EventArgs e)
{
    System.Web.Http.GlobalConfiguration.Configuration.EnableCors();
}
```

For more information about CORS, see [Configure CORS for WebAPI](#)

5. Add the following code to the **Application_Start** method to register the Routing configuration:

```
'csharp
protected void Application_Start(object sender, EventArgs e)
{
    System.Web.Http.GlobalConfiguration.Configuration.EnableCors();
    System.Web.Http.GlobalConfiguration.Configuration.Routes.MapHttpRoute(
        "
```

```

name: "DefaultApi",
routeTemplate: "api/{controller}/{action}/{id}",
defaults: new { id = RouteParameter.Optional });
}
`
```

For more information about routing tables, see [Routing in ASP.NET Web API](#).

Add Web API Controller

1. Right-click the project and select **Add > New Item** from the context menu.
2. In the Add New Item dialog, select **Web API Controller** class and name it as **ReportingAPIController**, and then click **Add**.

![Adding a new controller to the project](/static/assets/angular/report-designer/report-service/web-api-images/add-web-api-controller.png)

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Configure Web API

The **IReportDesignerController** interface contains the required actions and helper methods declaration to process the designer file and data actions. The **ReportDesignerHelper** and **ReportHelper** class contains methods that help to process Post or Get request from the control and return the response.

| Methods | Description | |
|--------------------------------------------------------------------------------------------------------------|-------------|--|
| ----- ----- | | |
| PostDesignerAction Action (HttpPost) method for posting the request for designer actions. | | |
| UploadReportAction Action (HttpPost) method for posted file actions. | | |
| SetData Write the resource into storage location. | | |
| GetData Read the resource from storage location. | | |
| GetImage Action (HttpGet) method for getting resource for report images. | | |
| PostReportAction Action (HttpPost) method for posting the request for report process. | | |
| GetResource Action (HttpGet) method for getting resource for report. | | |
| OnInitReportOptions Report initialization method that is triggered when the report begins to be processed. | | |
| OnReportLoaded Report loaded method that occurs when the report and sub report start loading. | | |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| Methods | Description | |
|-------------|-------------|--|
| ----- ----- | | |

| | |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

ReportHelper

The class `ReportHelper` contains helper methods that help process Post or Get request for report preview action and returns the response to the Report Designer. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

1. Open the `ReportingAPIController` and add the following using statement.

`csharp

```
using BoldReports.Web;  
using BoldReports.Web.ReportDesigner;  
using BoldReports.Web.ReportViewer;  
using System.IO;  
using System.Reflection;  
using System.Web;  
`
```

2. Next, add the `[EnableCors]` attribute to the `ReportingAPIController` class.

`csharp

```
[System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]

public class ReportingAPIController : ApiController
{
}

`
```

3. Inherit the `IReportDesignerController` interface, and implement its methods (replace the following code in newly created Web API controller).

`csharp

```
[System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
public class ReportingAPIController : ApiController, IReportDesignerController
```

```
{  
private string GetFilePath(string itemName, string key)  
{  
    string targetFolder = HttpContext.Current.Server.MapPath("~/");  
    targetFolder += "Cache";  
    if (!Directory.Exists(targetFolder))  
    {  
        Directory.CreateDirectory(targetFolder);  
    }  
    if (!Directory.Exists(targetFolder + "\\\" + key))  
    {  
        Directory.CreateDirectory(targetFolder + "\\\" + key);  
    }  
    return targetFolder + "\\\" + key + "\\\" + itemName;  
}  
[System.Web.Http.ActionName("GetImage")]  
[AcceptVerbs("GET")]  
public object GetImage(string key, string image)  
{  
    return ReportDesignerHelper.GetImage(key, image, this);  
}  
public object PostDesignerAction(Dictionary<string, object> jsonResult)  
{  
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null);  
}  
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)  
{  
    errorMessage = string.Empty;  
    if (itemData.Data != null)  
    {  
        File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);  
    }  
    else if (itemData.PostedFile != null)  
{  
}
```

```
{  
var fileName = itemId;  
if (string.IsNullOrEmpty(itemId))  
{  
    fileName = Path.GetFileName(itemData.PostedFile.FileName);  
}  
itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));  
}  
return true;  
}  
  
public ResourceInfo GetData(string key, string itemId)  
{  
    var resource = new ResourceInfo();  
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));  
    return resource;  
}  
  
public void UploadReportAction()  
{  
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);  
}  
[System.Web.Http.ActionName("GetResource")]  
[AcceptVerbs("GET")]  
public object GetResource(string key, string resourcetype, bool isPrint)  
{  
    //Returns the report resource for the requested key.  
    return ReportHelper.GetResource(key, resourcetype, isPrint);  
}  
  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");  
    reportOption.ReportModel.ExportResources.Scripts = new List<string>  
    {  
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",  
    }  
}
```

```
resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
//Chart component script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
//Gauge component scripts
resourcesPath + @"\bold-reports\common\ej2-base.min.js",
resourcesPath + @"\bold-reports\common\ej2-data.min.js",
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
    resourcesPath + @"\jquery-1.7.1.min.js"
};

}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

public object PostReportAction(Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this as IReportController);
}

`
```

4. Compile and run the Web API service application.

Create ASP.NET Core Web API Service

In this section, you will learn how to create a ASP.NET Core Web API for Report Designer using the new ASP.NET Core Web Application template.

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Designer API creation.

1. Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.
2. Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

![Creating a new ASP.NET Core Web Application Project](/static/assets/angular/report-designer/report-service/core-api-images/aspnet-new-project.png)

3. Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/angular/report-designer/report-service/core-api-images/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. In the Solution Explore, right-click the Dependencies, select Manage NuGet Packages and then search the package **BoldReports.Net.Core** and install to the application. The following provides detail of the packages and its usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core package**.

Newtonsoft.Json | Serialize and deserialize the data or report for report designer. It is a mandatory package for the report designer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer and designer. It should be referred in project when process the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0 .

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Designer NuGet packages.

Register CORS

Browser security prevents Report Designer from making requests to your API Service when both runs in a different domain. To allow access to your API service from a different domain, you must enable cross-origin requests.

Register `AddCors` in `Startup.ConfigureServices` to add CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
    {
        builder.AllowAnyOrigin()
            .AllowAnyMethod()
            .AllowAnyHeader();
    }));
    services.AddMvc();
}
'
```

For more information about CORS, see [Configure CORS for API](#)

Add API Service

1. Right-click the project and select **Add > New Item** from the context menu.
2. In the Add New Item dialog, select **API Controller** class and name it as `ReportingAPIController`, and then click **Add**.

![Adding a new controller to the project](/static/assets/angular/report-designer/report-service/core-api-images/add-core-api-controller.png)

While adding API Controller class, name it with the suffix `Controller` that is mandatory.

Configure Web API

The `IReportDesignerController` interface contains the required actions and helper methods declaration to process the designer file and data actions. The `ReportDesignerHelper` and `ReportHelper` class contains methods that help to process Post or Get request from the control and return the response.

| Methods | Description | |
|---------|-------------|-------|
| ----- | ----- | ----- |

| | |
|----------------------------------------------------------------------------------------------------------|--|
| PostDesignerAction Action (HttpPost) method for posting the request for designer actions. | |
| UploadReportAction Action (HttpPost) method for posted file actions. | |
| SetData Write the resource into storage location. | |
| GetData Read the resource from storage location. | |
| GetImage Action (HttpGet) method for getting resource for report images. | |
| PostReportAction Action (HttpPost) method for posting the request for report process. | |
| GetResource Action (HttpGet) method for getting resource for report. | |
| OnInitReportOptions Report initialization method that occurs when the report is about to be processed. | |
| OnReportLoaded Report loaded method that occurs when the report and sub report start loading. | |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| | | |
|----------------------------------------------------------------------|-------------|--|
| Methods | Description | |
| ----- ----- | | |
| GetResource Returns the report resource for the requested key. | | |
| ProcessReport Processes the report request and returns the result. | | |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request for report preview action and returns the response to the Report Designer. It has the following methods.

| | | |
|----------------------------------------------------------------------|-------------|--|
| Methods | Description | |
| ----- ----- | | |
| GetResource Returns the report resource for the requested key. | | |
| ProcessReport Processes the report request and returns the result. | | |

1. Open the **ReportingAPIController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
'
```

2. Add the **{action}** placeholder in the route template, if not contains with default controller attribute.

```
'csharp
```

```
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller
{
}
`
```

3. Next, add the `[EnableCors]` attribute to the `ReportingAPIController` class and specify the policy name which given in `Startup.ConfigureServices`.

```
`csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]

[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller,
BoldReports.Web.ReportDesigner.IReportDesignerController
{
}
`
```

4. Inherit the `IReportDesignerController` interface, and implement its methods (replace the following code in newly created Web API controller).

```
`csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]

[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller,
BoldReports.Web.ReportDesigner.IReportDesignerController
{
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;

    public ReportingAPIController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }

    private string GetFilePath(string itemName, string key)
    {
```

```
string targetFolder = this._hostingEnvironment.WebRootPath + "\\";
targetFolder += "Cache";
if (!System.IO.Directory.Exists(targetFolder))
{
    System.IO.Directory.CreateDirectory(targetFolder);
}
if (!System.IO.Directory.Exists(targetFolder + "\\\" + key))
{
    System.IO.Directory.CreateDirectory(targetFolder + "\\\" + key);
}
return targetFolder + "\\\" + key + "\\\" + itemName;
}

public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}

public object GetResource(ReportResource resource)
{
    return ReportHelper.GetResource(resource, this, _cache);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

[HttpPost]
public object PostDesignerAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null, this._cache);
}

public object PostFormDesignerAction()
{
```

```
return ReportDesignerHelper.ProcessDesigner(null, this, null, this._cache);
}

public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, this._cache);
}

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        System.IO.File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = System.IO.Path.GetFileName(itemData.PostedFile.FileName);
        }
        using (System.IO.MemoryStream stream = new System.IO.MemoryStream())
        {
            itemData.PostedFile.OpenReadStream().CopyTo(stream);
            byte[] bytes = stream.ToArray();
            var writePath = this.GetFilePath(fileName, key);
            System.IO.File.WriteAllBytes(writePath, bytes);
            stream.Close();
            stream.Dispose();
        }
    }
}
```

```
}

}

return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = System.IO.File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

[HttpPost]
public void UploadReportAction()
{
    ReportDesignerHelper.ProcessDesigner(null, this, this.Request.Form.Files[0], this._cache);
}
}
```

5. Compile and run the Web API service application.

How to section for Angular Report Designer

This section helps to get the answer for the frequently asked questions in Report Designer.

- [Getting Started for earlier version](#)
- [How to add datasource and dataset for Report Designer from application?](#)

Add Web Report Designer to an Angular application using Angular CLI

This section explains the steps required to add Web Report Designer to an Angular application.

If you are using higher version of [Bold Reports Angular](#) (>= v2.2.28), then refer [Getting Started for latest version](#).

Prerequisites

Before getting started with Bold Report Designer, make sure your development environment includes the following,

- [Node JS](#) (version 8.x or 10.x)
- [NPM](#) (v3.x.x or higher)

Install the Angular CLI

Angular provides the easiest way to set Angular CLI projects using [Angular CLI](#) tool. To install the CLI application globally to your machine, run the following command in Command Prompt.

```
'typescript  
npm install -g @angular/cli  
'
```

To learn more about angular-cli commands refer [here](#)

Create a new application

To create a new Angular application run the below command in Command Prompt.

```
'typescript  
ng new project-name  
E.g : ng new reportdesignerapp  
'
```

The `ng new` command prompts you for information about features to include in the initial app project. Add the Angular Routing to your angular application by entering `y` in the prompt window then press **Enter** key. Now, choose the CSS stylesheet format using the arrow keys and then press **Enter** key. The Angular CLI installs the required Angular npm packages and other dependencies.

Configure Bold Report Designer in Angular CLI

Bold Reporting packages are distributed in npm as [@boldreports/angular-reporting-components](#).

1. To configure the web Report Designer component, change the directory to your application's root folder.

```
'typescript  
cd project-name  
E.g : cd reportdesignerapp  
'
```

2. Run the below commands, to install the internal dependencies.

```
'typescript  
npm install @boldreports/angular-reporting-components --save  
'
```

3. Install the `typings` dependencies `reports.all`, `jquery` and `jsrender` using below commands.

```
'typescript  
npm install @boldreports/types
```

Add Web Report Designer to an Angular application using Angular CLI | Configure Bold Report Designer in Angular CLI

```
npm install --save-dev @types/jquery  
npm install --save-dev @types/jsrender  
`
```

4. Add the `jquery`, `jsrender` and `reports.all` under `types` node in `tsconfig.app.json` file.

```
'js  
{  
  "extends": "../tsconfig.json",  
  "compilerOptions": {  
    "outDir": "./out-tsc/app",  
    "types": [  
      "jquery",  
      "jsrender",  
      "reports.all"  
    ]  
  },  
  "exclude": [  
    "test.ts",  
    "/**.spec.ts"  
  ]  
}
```

5. Report Designer requires `window.jQuery` object to render the component. Import jQuery in `src/polyfills.ts` file as in the following code snippet.

```
'js  
import * as jquery from 'jquery';  
window['jQuery'] = jquery;  
window['$'] = jquery;  
`
```

6. Register the `@boldreports/types` under the `typeRoots` node in the `tsconfig.json` file.

![Css script reference](/static/assets/angular/report-designer/images/register-typings.png)

Refer the already configured import codes in our [webpack angular seed](#) application.

Adding CSS reference

1. Open the `angular.json` file from your application's root directory.
2. Refer the web Report Designer component style scripts `bold.reports.all.min.css` and `bold.reportdesigner.min.css` under the `styles` node of `projects` section as in the below code example.

If you are using Angular 6 or lower version project, add the changes in `angular-cli.json` file.

```
![Css script reference](/static/assets/angular/report-designer/images/Getting-Started_img1.png)
`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "reportdesignerapp": {
      "root": "",
      "sourceRoot": "src",
      "projectType": "application",
      "prefix": "app",
      "schematics": {},
      "architect": {
        "build": {
          "builder": "@angular-devkit/build-angular:browser",
          "options": {
            "outputPath": "dist/reportdesignerapp",
            "index": "src/index.html",
            "main": "src/main.ts",
            "polyfills": "src/polyfills.ts",
            "tsConfig": "src/tsconfig.app.json",
            "assets": [
              "src/favicon.ico",
              "src/assets"
            ],
            "styles": [

```

```

    "src/styles.css",
    "./node_modules/@boldreports/javascript-reporting-
    controls/Content/material/bold.reports.all.min.css",
    "./node_modules/@boldreports/javascript-reporting-
    controls/Content/material/bold.reportdesigner.min.css"
],
"scripts": [],
"es5BrowserSupport": true
},
...
...
}
`
```

In the above code snippet the `material` theme is used. You can modify the theme based on your application, refer the following syntax: `./node_modules/@boldreports/javascript-reporting-controls/Content/[theme-name]/bold.reports.all.min.css` and `./node_modules/@boldreports/javascript-reporting-controls/Content/[theme-name]/bold.reportdesigner.min.css`

[Adding code mirror reference](#)

Report Designer requires the code mirror scripts and styles to edit the SQL queries and Visual Basic code functions with syntax highlighter.

1. Run the below command, to install the code mirror dependencies.

```

`typescript
npm install codemirror
`
```

2. Open the `angular.json` file from your application's root directory and Refer the code mirror script and styles under the `scripts` and `styles` node of `projects` section as in the below code example.

```

![Css script reference](/static/assets/angular/report-designer/images/code-mirror-references.png)
`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
```

```
"projects": {  
  "reportdesignerapp": {  
    "root": "",  
    "sourceRoot": "src",  
    "projectType": "application",  
    "prefix": "app",  
    "schematics": {},  
    "architect": {  
      "build": {  
        "builder": "@angular-devkit/build-angular:browser",  
        "options": {  
          "outputPath": "dist/reportdesignerapp",  
          "index": "src/index.html",  
          "main": "src/main.ts",  
          "polyfills": "src/polyfills.ts",  
          "tsConfig": "src/tsconfig.app.json",  
          "assets": [  
            "src/favicon.ico",  
            "src/assets"  
          ],  
          "styles": [  
            "src/styles.css",  
            "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css",  
            "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css",  
            "./node_modules/codemirror/lib/codemirror.css",  
            "./node_modules/codemirror/addon/hint/show-hint.css"  
          ],  
          "scripts": [  
            "./node_modules/codemirror/lib/codemirror.js",  
            "./node_modules/codemirror/addon/hint/show-hint.js",  
            "./node_modules/codemirror/addon/hint/sql-hint.js",  
            "./node_modules/codemirror/mode/vb/vb.js"  
          ]  
        }  
      }  
    }  
  }  
}
```

```
],
"es5BrowserSupport": true
},
...
...
}
`
```

Adding Scripts reference

To render chart data visualization item in angular bold report designer, need to refer the following scripts in your angular application.

```
`basic
bold.reports.common.min.js
bold.reports.widgets.min.js
bold.report-designer-widgets.min.js
ej.chart.min.js
`
```

1. Open the `angular.json` file from your application's root directory.
2. Refer above mentioned list of scripts under the `scripts` node of `projects` section as in the below code example.

If you are using Angular 6 or lower version project, add the changes in `angular-cli.json` file.

![Css script reference](/static/assets/angular/report-designer/images/script-reference.png)

```
`js
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "reportdesignerapp": {
      "root": "",
      "sourceRoot": "src",
      "projectType": "application",
      "prefix": "app",
      "schematics": {}
    }
  }
}`
```

```
"architect": {  
  "build": {  
    "builder": "@angular-devkit/build-angular:browser",  
    "options": {  
      "outputPath": "dist/reportdesignerapp",  
      "index": "src/index.html",  
      "main": "src/main.ts",  
      "polyfills": "src/polyfills.ts",  
      "tsConfig": "src/tsconfig.app.json",  
      "assets": [  
        "src/favicon.ico",  
        "src/assets"  
      ],  
      "styles": [  
        "src/styles.css",  
        "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css",  
        "./node_modules/@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css",  
        "./node_modules/codemirror/lib/codemirror.css",  
        "./node_modules/codemirror/addon/hint/show-hint.css"      ],  
      "scripts": [  
        "./node_modules/codemirror/lib/codemirror.js",  
        "./node_modules/codemirror/addon/hint/show-hint.js",  
        "./node_modules/codemirror/addon/hint/sql-hint.js",  
        "./node_modules/codemirror/mode/vb/vb.js",  
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/bold.reports.common.min.js",  
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/bold.reports.widgets.min.js",  
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/common/bold.report-designer-widgets.min.js",  
        "./node_modules/@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min.js"  
      ],  
      "es5BrowserSupport": true  
    }  
  }  
}
```

},

...

...

}

,

Adding Report Designer component

To add the web Report Designer component refer the following steps:

1. Open the `src/app/app.module.ts` file.
2. Import `BOLDREPORTDESIGNERCOMPONENTS` from `@boldreports/angular-reporting-components/src/reportdesigner.component` package as in the below code snippet.

`'typescript'`

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { BOLDREPORTDESIGNERCOMPONENTS } from '@boldreports/angular-reporting-
components/src/reportdesigner.component';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent,
    BOLDREPORTDESIGNERCOMPONENTS
  ],
  imports: [
    BrowserModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

3. Open the `src/app/app.component.html` file and initialize the web Report Designer.
4. You can replace the following code snippet in the `app.component.html` file.

`'html'`

```
<bold-reportdesigner id="designer" style="position: absolute; height: 550px; width: 1250px;">
</bold-reportdesigner>
`
```

5. Open the `src/app/app.component.ts` and replace the below code example.

```
'typescript
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'reportdesignerapp';
  public serviceUrl: string;
  constructor() {
    // Initialize the Report Designer properties here.
  }
}
`
```

If you have faced the issue 'ej' is not defined after the above configuration in Angular CLI latest version 7, Please refer the below code snippet in your application where you have rendered Bold Reporting Components(model file) to resolve the issue.

```
'typescript
/// <reference types="reports.all" />
`
```

Create Web API service

The web Report Designer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API service.

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

Set Web API service URL

To set Web API service, open the `app.component.ts` file and add the code snippet as in the constructor.

`'typescript'`

```
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'reportdesignerapp';
  public serviceUrl: string;
  constructor() {
    this.serviceUrl = "https://demos.boldreports.com/services/api/ReportingAPI";
  }
}
`
```

Open the `app.component.html` to set `ServiceUrl` property of web Report Designer as in the following code snippet.

`'javascript'`

```
<bold-reportdesigner id="designer" [serviceUrl] = "serviceUrl" style="position: absolute; height: 550px; width: 1250px;">
</bold-reportdesigner>
`
```

Serve the application

1. Go to the workspace folder (`reportdesignerapp`).
2. Launch the server by using the CLI command `ng serve`, with the `--open` option.

`'typescript'`

```
ng serve --open
`
```

3. The `ng serve` command launches the server and the `--open` option automatically opens your browser to `http://localhost:4200/`

![report designer output](/static/assets/angular/report-designer/images/Getting-Started_img9.png)

See Also

[Production deployment](#)

How to add the data source and dataset for Report Designer from application

You can add the data for reports in the Report Designer from application level on initializing the Report Designer. This can be achieved using the `addDataSource` and `addDataSet` functions, by which you can add the data source and dataset for the reports at the time of initialization of Report Designer.

Find the following steps to add the data source and dataset for Report Designer from application.

- Create a function and bind it with the `create` API as shown in the following code snippet.

```
'html
<bold-reportdesigner id="designer"
[serviceUrl]="serviceUrl"
(create)="controlInitialized()"

</bold-reportdesigner>
`
```

```
'typescript
export class AppComponent {
...
public controlInitialised(){
...
}
`
```

- Use the `addDatasource` method to add the data source in Report Designer as as shown in the following code snippet,

```
'typescript
export class AppComponent {
title = 'reportdesignerapp';
public serviceUrl: string;
public datasource: any =
```

```
{
  type:'BoldReports.RDL.DOM.DataSource',
  Name:'DataSource1',
  Transaction:false,
  DataSourceReference:null,
  SecurityType:'DataBase',
  ConnectionProperties:{
    type:'BoldReports.RDL.DOM.ConnectionProperties',
    ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
    EmbedCredentials:false,
    DataProvider:'SQL',
    IsDesignState:false,
    IntegratedSecurity:false,
    UserName:'',
    PassWord:'',
    Prompt:'Specify the Username and password for DataSource DataSource1',
    CustomProperties: []
  }
};

public controlInitialised(){
  var designerObj = $('#designer').data('boldReportDesigner');
  designerObj.addDataSource(datasource);
}

}
`
```

- Use the `addDataSet` method to add dataset in Report Designer as shown in the following code snippet,

```
`typescript
export class AppComponent {
  title = 'reportdesignerapp';
  public serviceUrl: string;
  public dataset: any =
{`
```

```
type:'BoldReports.RDL.DOM.DataSet',
Name:'DataSet1',
Fields:[
{ type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},
{ type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},
{ type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },
{ type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null}
],
Query: {
type: "BoldReports.RDL.DOM.Query",
CommandText: "SELECT
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[HumanResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM
[HumanResources].[Department]",
CommandType: 0,
DataSourceName: "DataSource1",
QueryDesignerState: {
type: "BoldReports.RDL.DOM.QueryDesignerState",
Expressions: null,
Filters: null,
Joins: null,
StoredProcedure: null,
Tables: [
{
type: "BoldReports.RDL.DOM.Table",
Columns: [
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "DepartmentID" },
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "Name" }
]
```

```
},  
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
IsSelected: true, Name: "GroupName"  
},  
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
IsSelected: true, Name: "ModifiedDate"  
}  
],  
Name: "Department",  
Schema: "HumanResources",  
SchemaLevels: [  
{ Name: "HumanResources", SchemaType: "Schema"},  
{ Name: "Tables", SchemaType: "Category"},  
{ Name: "Department", SchemaType: "Table"}  
]  
}  
]  
},  
QueryParameters: [],  
Timeout: 0  
},  
CaseSensitivity:0,  
Collation:null,  
AccentSensitivity:0,  
KanatypeSensitivity:0,  
WidthSensitivity:0,  
Filters:[],  
SharedDataSet:null,  
InterpretSubtotalsAsDetails:0,  
DataSetInfo:null,  
DataSetObject:null  
};  
public controlInitialised(){
```

```

var designerObj = $('#designer').data('boldReportDesigner');
designerObj.addDataSet(dataset);
}
}
'

```

Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

Breaking Changes

When you upgrade from previous versions of Bold Reports to 2.2.23, there are few breaking changes. The following section explains the breaking changes for Bold Reports version 2.2.23.

Designer resource read and write API

The resource write and read API's in `IReportDesignerController` are modified to simplify the resource write and read actions with Bold Report Designer.

Below are the new API details tabulated against old API's.

| Old API | New API | Description |
|-------------|---------|-----------------------------------------------------|
| UploadFile | SetData | Writes the designer resource into storage location. |
| GetFilePath | GetData | Reads the designer resource from storage location. |
| GetFiles | | |
| GetFile | | |

Parameters

SetData

| Parameter Name | Type | Description |
|----------------|---------------|--------------------------------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |
| itemData | object | Gets the resource data. |
| | Property Name | |
| | Data | byte array Gets the resource data as byte array. To write |

| | | | | |
|--------------|------------|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|
| | | | an external resource into storage location, pass the resource data in this property. | |
| | PostedFile | HttpPostedFile | Gets the uploaded resource data as HttpPostedFile . The resource data uploaded within report designer will be received in this property. | |
| errorMessage | string | | | Returns the error message, if the write action is failed. |

GetData

| Parameter Name | Types | Description |
|----------------|--------|-------------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |

Return Type

| API Name | Return Type | |
|---------------|-------------|----------------------------------------------------------|
| SetData | boolean | |
| GetData | object | |
| Property Name | Type | Description |
| Data | byte array | Contains the requested resource data as byte array. |
| errorMessage | string | Returns the error message, if the read action is failed. |

Code snippet

| Old snippet | New snippet |
|----------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| UploadFilecsharp public bool UploadFile(System.Web.HttpPostedFile | SetDatacsharp public bool SetData(string key, string itemId, ItemInfo itemData, out string |

| | |
|---------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| <code>httpPostedFile){ //Implement resource write actions}</code> | <code>errorMessage){ //Implement resource write actions}</code> |
| <code>GetFiles csharppublic List GetFiles(FileType fileType){ //Implement resource read actions}</code> | |
| <code>GetFilePath csharppublic string GetFilePath(string fileName){ //Implement actions to get file path}</code> | <code>GetData csharppublic ResourceInfo GetData(string key, string itemId){ //Implement resource read actions}</code> |
| <code>GetFile csharppublic FileModel GetFile(string filename, bool isOverride){ //Implement resource read actions}</code> | |

Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

Breaking Changes

To support Angular 9, we made the below, breaking changes with Bold Reports Angular components version 2.2.28.

| Scenario | Existing Angular Package | Latest Angular Package |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Importing components | <code>// imports Report viewerimport { BOLDREPORTVIEWERCOMPONENTS } from â€˜@boldreports/angular-reporting-components/src/reportviewer.componentâ€™; // imports Report Designerimport { BOLDREPORTDESIGNERCOMPONENTS } from â€˜@boldreports/angular-reporting-components/src/reportdesigner.componentâ€™;</code> | <code>// imports Report viewerimport { BoldReportViewerModule } from â€˜@boldreports/angular-reporting-componentsâ€™; // imports Report Designerimport { BoldReportDesignerModule } from â€˜@boldreports/angular-reporting-componentsâ€™;</code> |
| Importing modules | <code>// imports both Report Viewer and Designerimport { BoldReportsAngularModule } from â€˜@boldreports/angular-reporting-componentsâ€™;</code> | <code>// imports both Report Viewer and Designerimport { BoldReportsModule } from â€˜@boldreports/angular-reporting-componentsâ€™;</code> |
| Script reference | <code>We only refer the data-visualization scripts.Example:import â€˜@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.minâ€™;import â€˜@boldreports/javascript-reporting-</code> | <code>We need to refer the Report Viewer and Designer scripts along with the existing data-visualization script references.Report Viewerimport â€˜@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.minâ€™;Report Designerimport</code> |

| | | |
|--|-------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | controls/Scripts/data-visualization/ ej2-circulargauge.min; | â€˜@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.minâ€™;Data visualizationExample:import â€˜@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.minâ€™;import â€˜@boldreports/javascript-reporting-controls/Scripts/data-visualization/ ej2-circulargauge.min; |
|--|-------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Reporting tools for JavaScript

Enterprise-class reporting tools for JavaScript development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your JavaScript applications.

How to best read this user guide

The best way to get started would be to read the Getting Started section of the documentation for the control that you would like to start using first. The Getting Started guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Another valuable resource is the API reference which provides detailed information on the object hierarchy as well as the settings available on every object.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Reporting tools for JavaScript

Enterprise-class reporting tools for JavaScript development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your JavaScript applications.

How to best read this user guide

The best way to get started would be to read the Getting Started section of the documentation for the control that you would like to start using first. The Getting Started guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Another valuable resource is the API reference which provides detailed information on the object hierarchy as well as the settings available on every object.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

System Requirements

This topic describes the software and hardware requirements for setting up the development environment of Bold Reports Javascript.

Supported Operating Systems

- Windows 7+, 8+
- Windows Server 2008 R2+

Hardware Requirements

The following hardware requirements are necessary setting up the development environment of Bold Reports Javascript:

- 1 GHZ or faster, 32 bit or 64 bit processor.
- 1 GB RAM for 32 bit or 2 GB RAM for 64 bit.

Software Requirements

The following software requirements are necessary setting up the development environment of Bold Reports Javascript:

- Microsoft Visual Studio 2010 or later
- Internet Information Services (IIS) 7.0+

Browser Compatibility

- IE 9+
- Microsoft Edge
- Mozilla Firefox 22+
- Chrome 17+
- Opera 12+
- Safari 5+

See Also

- [Licensing procedure for deployment](#)

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major

capabilities of RDL reporting and export the report to PDF, Excel, CSV, Word, PowerPoint, and HTML formats. Some of the key features are,

- Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.
- Easily customize each element of report viewer and provide events for report processing customization.
- Supports jQuery, Angular, React, Ember, Aurelia, PHP, and JSP.

Display ssrs rdl report in Bold Reports HTML5 JavaScript Report Viewer

This section explains you the steps required to create your first JavaScript reporting application to display already created SSRS RDL report in Bold Reports HTML5 JavaScript Report Viewer without using a Report Server.

HTML file creation

Create a basic HTML file as shown below and place it in a separate folder.

```
'html
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Report Viewer first HTML page</title>
</head>
<body>
</body>
</html>
'
```

Refer scripts and CSS

Directly refer all the required scripts and style sheets from [CDN](#) links that are mandatorily required to use the Report Viewer as in the following order.

- bold.reports.all.min.css
- jquery-1.10.2.min.js
- bold.reports.common.min.js
- bold.reports.widgets.min.js
- ej.chart.min.js - Renders the chart item. Add this script, only if your report contains the chart report item.
- ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js - Render the gauge item. Add these scripts only if your report contains the gauge report item.
- ej.map.min.js - Renders the map item. Add this script only if your report contains the map report item.
- bold.report-viewer.min.js

Refer to the [Bold Reports CDN](#) to learn more details about Bold Reports CDN scripts and style sheets links.

You can replace the following code in `<head>` tag of the Report Viewer HTML page.

Whether you want to get the scripts and style sheets as local, then install the `BoldReports.Javascript` NuGet package in your application.

```
'html
<link
href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet"
/>

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>

<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>

<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>

<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>

<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

To learn more about rendering a report with data visualization report items, refer to the [how to render data visualization report items](#) section.

Adding Report Viewer Widget

Add the `<div>` element within the `<body>` section, which acts as a container for `boldReportViewer` widget to render and then initialize the `boldReportViewer` widget within the script section as shown below,

Display ssrs rdl report in Bold Reports HTML5 JavaScript Report Viewer service Set report path and Web API

```
'html
<div style="height: 600px; width: 950px;">
    <!-- Creating a div tag which will act as a container for boldReportViewer widget.-->
    <div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>
    <!-- Setting property and initializing boldReportViewer widget.-->
    <script type="text/javascript">
        $(function () {
            $("#viewer").boldReportViewer();
        });
    </script>
</div>
'
```

Create Web API service

The Report Viewer requires a Web API service to process the report files. You can skip this step and use the online [Web API services](#) to preview the already available reports or you should create any one of the following Web API services:

- [ASP.NET Web API Service](#)
- [ASP.NET Core Web API Service](#)

Adding already created report

If you have created a new service, you can add the reports from the Bold Reports installation location. For more information, refer to [samples and demos](#) section.

1. Create a folder **Resources** in your Web API application to store RDL reports and add the already created reports to it.
2. Add already created reports to the newly created folder.

In this tutorial, the **sales-order-detail.rdl** report is used, and it can be downloaded in this [link](#).

To create new report refer to the [create RDL report](#) section.

Set report path and Web API service

To render the reports available in your application, set the [reportPath](#) and [reportServiceUrl](#) properties of the Report Viewer.

```
'html
<div style="height: 600px; width: 950px;">
    <!-- Creating a div tag which will act as a container for boldReportViewer widget.-->
    <div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>
    <!-- Setting property and initializing boldReportViewer widget.-->
```

```
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "https://demos.boldreports.com/services/api/ReportViewer",
        reportPath: '~/Resources/docs/sales-order-detail.rdl'
    });
});
</script>
</div>
`
```

In the above code, the `sales-order-detail.rdl` report and `reportServiceUrl` used from online URL. You can view the already created Web API service from the [Reporting Service](#) git hub location. For more information, see [samples and demos](#) section.

Preview the report

Open your HTML page in the web browser and the Report Viewer will display the report as shown below.

```
{% tab demoPath="javascript/report-viewer/getting-started/initialize-report-viewer"
files="index.html,index.js,ReportViewController.cs" %}

{% tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% tabItem header="ReportViewController.cs" %}

`csharp
public class ReportViewController : ApiController, IReportController
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }
}
```

```
}

// Get action for getting resources from the report
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]

public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

// Method that will be called when initialize the report options before start processing the report
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    // You can update report options here
}

// Method that will be called when reported is loaded
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    // You can update report options here
}

}

` 

{%
endtabItem %}

{%
endtab %}
```

See Also

[Render report with data visualization report items](#)

[Render Report Server reports](#)

[Create RDLC report](#)

[Render RDLC reports](#)

[Preview report in print mode](#)

[Set data source credential for shared data sources](#)

[Change data source connection string](#)

[List of SSRS server versions are supported in Bold Reports](#)

Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports set the `reportServerUrl`, `reportPath`, and `reportServiceUrl` properties as in the following code snippet.

```
'js
<div style="height: 100%; width: 100%;">
  <div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>
  <script type="text/javascript">
    $(function () {
      $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/SSRSReports",
        reportPath: "/SSRSSamples/Territory Sales",
        reportServerUrl: "http://<servername>/reportserver$instanceName"
      });
    });
  </script>
</div>
'
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of `/folder name/report name`.

Network credentials for SSRS

The network credentials are required to connect with the specified SSRS Report Server using Report Viewer. Specify the `ReportServerCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
  //Add SSRS Report Server credential
  reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
  "RDLReport1");
}
```

Set data source credential for shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS server. If the report has any data source that uses credentials to connect with the database, then you must specify the `DataSourceCredentials` for each report data source to establish database connection.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Data source credentials must be added for shared data sources that do not have credentials in the connection strings.

Build and run the application. Preview and edit the result using the following.

'js

```
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "https://demos.boldreports.com/services/api/SSRSDataSourceCredentials",
        reportPath: "/SSRSSamples/Territory Sales",
        reportServerUrl: "http://<servername>/reportserver$instanceName"
    });
});
```

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-demo",
    "N@c)=Y8s*1&dh"));
}
```

```
}
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

Change data source connection string

You can change the connection string of a report data source before it is loaded in the Report Viewer. The `DataSourceCredentials` class provides the option to set and update the modified connection string as in the following code snippet.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.DataSourceCredentials.Add(new
        BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-
        demo", "N@c)=Y8s*1&dh", "Data Source=dataplatformdemodata.syncfusion.com;Initial
        Catalog=AdventureWorks"));
}
```

The previous code shows an option to change the connection string only, but the class provides multiple options to change data source information. To learn more about this, refer to the `DataSourceCredentials` class.

Render linked reports

You can render a linked report that point to an existing report, which is published in the SSRS Report Server. Also, it is possible to set the parameter, data source, credential, and other properties as like normal SSRS reports using the Report Viewer.

```
'js
```

```
<div style="height: 100%; width: 100%;">
<div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/SSRSReports",
        reportPath: "/SSRSSamples/Territory Sales_Link",
        reportServerUrl: "http://<servername>/reportserver$instanceName"
    });
});
</script>
```

```
</div>
```

The `Territory Sales_Link` is a linked report created for `Territory Sales` report that is already available on the SSRS Report Server.

Load SharePoint Server reports

To render SharePoint server reports set the `reportServerUrl`, `reportPath`, and `reportServiceUrl` properties as in the following code snippet.

```
'js
<div style="height: 100%; width: 100%;">
  <div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>
  <script type="text/javascript">
    $(function () {
      $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/SharePointReports",
        reportPath: "http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl",
        reportServerUrl: "http://<servername>/reportserver$instanceName"
      });
    });
  </script>
</div>
'
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

In SharePoint integrated mode, the `reportServerUrl` will be same as your site URL. The `reportPath` is relative to the Report Server URL with the file extension.

Forms credential for SharePoint server

The Forms credentials are required to connect with the specified SharePoint integrated SSRS Report Server using the Report Viewer. Specify the `ReportServerFormsCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
  //Add ReportServerFormsCredential for server
  reportOption.ReportModel.ReportServerFormsCredential = new
  BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");
```

Load SharePoint Server reports

Set data source credential for shared data sources

}

Set data source credential for shared data sources

The shared data source credentials can be added to the `DataSourceCredentials` property to connect with the database.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential and data source credentials
    reportOption.ReportModel.ReportServerFormsCredential = new
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Data source credentials must be added for shared data sources that do not have credentials in the connection strings.

Build and run the application. Preview and edit the result using the following.

'js

```
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "https://demos.boldreports.com/services/api/SharePointReports",
        reportPath: "http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl",
        reportServerUrl: "http://<servername>/reportserver$instanceName"
    });
});
```

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential and data source credentials
    reportOption.ReportModel.ReportServerFormsCredential = new
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Load Bold Report Server reports

Set data source credential for shared data sources

}

Load Bold Report Server reports

You can render the Bold Report Server reports in the Report Viewer easily without creating a Web API service. Bold Report Server provides the built-in Web API service that helps you to display the server reports.

The Report Viewer requires the [serviceAuthorizationToken](#) to connect and download the items from the Bold Report Server. Create a token for the user by using the server RESTful API, the following code is used to generate the token.

```
'js
<script type="text/javascript">
$(function () {
var dataValue = "";
var apiRequest = new Object();
apiRequest.password = "demo";
apiRequest.userid = "guest@boldreports.com";
$.ajax({
type: "POST",
url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
data: apiRequest,
success: function (data) {
dataValue = data.Token;
var token = JSON.parse(dataValue);
// Report Viewer initialization.
}
});
});
});
</script>
'
```

Set the Bold Report Server built-in service URL to the `reportServiceUrl` property. Assign the created token to [serviceAuthorizationToken](#) and `reportPath` to a report deployed on the server. You can use the following complete code in your HTML page.

```
'js
```

Load Bold Report Server reports

Set data source credential for shared data sources

```
<script type="text/javascript">
$(function () {
var dataValue = "";
var apiRequest = new Object();
apiRequest.password = "demo";
apiRequest.userid = "guest@boldreports.com";
$.ajax({
type: "POST",
url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
data: apiRequest,
success: function (data) {
dataValue = data.Token;
var token = JSON.parse(dataValue);
$("#viewer").boldReportViewer(
{
reportServiceUrl: "https://on-premise-demo.boldreports.com/reporting/reportservice/api/Viewer",
serviceAuthorizationToken: token["tokentype"] + " " + token["accesstoken"],
reportPath: '/Sample Reports/Company Sales',
ajaxBeforeLoad: "onAjaxRequest"
});
}
});
});
});

function onAjaxRequest(args) {
args.headers.push({
Key: 'serverurl', Value: 'https://on-premise-demo.boldreports.com/reporting/api/site/site1'
});
}

</script>
```

You can also load the report using Guid instead of report location. Set the Guid of the report in reportPath as like as reportPath: '91f24bf1-e537-4488-b19f-b37f77481d00'.

Build and run the application. Preview and edit the result using the following.

```
{% tab demoPath="javascript/report-viewer/reportserver-report" files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

Render RDLC report

The data binding support, allows you to view RDLC reports that exist on the local file system with JSON array and custom business object data collection. The following steps demonstrates how to render a RDLC report with JSON array and custom business object data collection.

Add the RDLC report `product-list.rdlc` from Bold Reports installation location to your application `Resources` folder. For more information, see [Samples and demos](#).

Bind data source at client side

Set the RDLC report path to the `reportPath` property.

Assign the `processingMode` property to `ProcessingMode.Local`.

Bind the JSON array collection to the `dataSources` property as shown in following code.

```
`js
<div style="height: 100%; width: 100%;">
<div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/ReportViewer",
        processingMode: ej.ReportViewer.ProcessingMode.Local,
        reportPath: '~/Resources/docs/product-list.rdlc',
        dataSources: [
            {
                value: [
                {

```

```

ProductName: "Baked Chicken and Cheese", OrderId: "323B60", Price: 55, Category: "Non-Veg",
Ingredients: "Grilled chicken, Corn and Olives.", ProductImage: ""

},
{

ProductName: "Chicken Delite", OrderId: "323B61", Price: 100, Category: "Non-Veg", Ingredients:
"Cheese, Chicken chunks, Onions & Pineapple chunks.", ProductImage: ""

},
{

ProductName: "Chicken Tikka", OrderId: "323B62", Price: 64, Category: "Non-Veg", Ingredients: "Onions,
Grilled chicken, Chicken salami & Tomatoes.", ProductImage: ""

}],
name: "list"
}]
});
});
});
</script>
</div>
`
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location.
For more information, see [Samples and demos](#).

Build and run the application to view the output result.

```

{%
tab demoPath="javascript/report-viewer/rdlc-report/bind-data-at-client" files="index.html,index.js"
%}

{%
tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{%
endtabItem %}

{%
endtab %}
```

Bind data source in Web API controller

The following steps help you to configure the Web API to render the RDLC report with business object data collection.

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
```

```
Category = "Non-Veg",
Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};
datas.Add(data);
return datas;
}
}

`
```

Set the `ProcessingMode` to `ProcessingMode.Local` and `ReportPath` to the RDLC report location.

Bind the business object data values collection by adding new item to the `DataSources` as in the following code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;
    reportOption.ReportModel.ReportPath =
        System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/docs/product-list.rdlc");
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
`
```

Here the **Name** is case sensitive and it should be same as in the data source name in the report definition.

The **Value** accepts **IList**, **DataSet**, and **DataTable** inputs.

Load report as stream

To load report as a stream, create a report stream using the **FileStream** class and assign the report stream to the **Stream** property.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string filePath = System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/docs/product-list.rdlc");
    // Opens the report from application Resources folder using FileStream
    FileStream reportStream = new FileStream(filePath, FileMode.Open, FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list", Value = ProductList.GetData() });
}
```

In the above code, **product-list.rdlc** report is loaded from the **Resources** folder location.

View Report Click

You can get the user selected parameter details when the user clicks on the **ViewReport** button in the parameter block. The **viewReportClick** event lets you to handle the **ViewReport** button click at client-side as in the following code.

'csharp

```
<script type="text/javascript">
$(function () {
    $("#container").boldReportViewer({
        reportServiceUrl: "/api/ReportViewer",
        reportPath: '~/Resources/docs/sales-order-detail.rdl',
        viewReportClick:"onViewReportClick"
    });
    function onViewReportClick(args) {
        args[0] = ({ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'], nullable: false });
    }
})
```

```
</script>
```

The model property in the event argument has the details of current processing report model.

Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

Add the sub report and main reports to your application **Resources** folder. In this tutorial, using the already created reports. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section for creating new reports.

Download the **side-by-side-main-report.rdl**, **side-by-side-sub-report.rdl** reports from [here](#). Also, you can add the report from Syncfusion installation location. For more information, see [Samples and demos](#). The reports used from installed location, requires **NorthwindIO_Reports.sdf** database to run, so add it to your application.

Set the **reportPath** and **reportServiceUrl** properties of the Report Viewer as in following code snippet.

```
'js
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/ReportViewer",
        reportPath: '~/Resources/docs/side-by-side-main-report.rdl'
    });
});
</script>
`
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

Build and run the application. Preview and edit the result using the following.

```
{% tab demoPath="javascript/report-viewer/subreport/preview-subreport" files="index.html,index.js"
%}
{% tabItem header="index.html" %}
`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
```

```
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
` 

{%- endtabItem %}

{%- endtab %}
```

Change subreport path

To change the subreport file path, set the **ReportPath** property of SubReportModel in the **OnInitReportOptions** method.

```
'csharp

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.ReportPath =
        System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/docs/sub-report-detail.rdl");
    }
}
`
```

Set subreport parameter

You can change the parameter default values of a subreport in the Web API Controller **OnReportLoaded**, method as given in the following code snippet.

```
'csharp

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.Parameters = new BoldReports.Web.ReportParameterInfoCollection();
        reportOption.SubReportModel.Parameters.Add(new BoldReports.Web.ReportParameterInfo()
        {
            Name = "SalesPersonID",
            Values = new List<string>() { "2" }
        });
    }
}
```

Render subreport

Modify subreport data source connection string

}

Modify subreport data source connection string

You can change the credential and connection information of the data sources used in the subreport using the SubReportModel in **OnInitReportOptions** method.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSourceCredentials = new
List<BoldReports.Web.DataSourceCredentials>();

        reportOption.SubReportModel.DataSourceCredentials.Add(new
BoldReports.Web.DataSourceCredentials("NorthWind", "Data
Source=dataplatformdemodata.syncfusion.com;Initial Catalog=Northwind;user id=demoreadonly@data-
platform-demo;password=N@c)=Y8s*1&dh"));

    }
}
```

Set subreport data source

You can bind local business object data source collection only for RDLC reports. To specify data source of a RDLC subreport, set the **ReportDataSource** property in **OnReportLoaded** method.

The RDL report has the connection information in report definition itself, so no need to bind data source.

Add the RDLC sub report and main reports to your application **Resources** folder. You can download it from [here](#).

Set the **reportPath** and **reportServiceUrl** properties of the Report Viewer as in following code snippet.

'js

```
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/SubreportDataSources",
        processingMode: ej.ReportViewer.ProcessingMode.Local,
        reportPath: '~/Resources/docs/product-list-main.rdlc'
```

```
});  
});  
</script>  
'
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
`csharp  
public class ProductList  
{  
    public string ProductName { get; set; }  
    public string OrderId { get; set; }  
    public double Price { get; set; }  
    public string Category { get; set; }  
    public string Ingredients { get; set; }  
    public string ProductImage { get; set; }  
    public static IList GetData()  
    {  
        List<ProductList> datas = new List<ProductList>();  
        ProductList data = null;  
        data = new ProductList()  
        {  
            ProductName = "Baked Chicken and Cheese",  
            OrderId = "323B60",  
            Price = 55,  
            Category = "Non-Veg",  
            Ingredients = "grilled chicken, corn and olives.",  
            ProductImage = ""  
        };  
        datas.Add(data);  
        data = new ProductList()  
        {
```

Render subreport

Set subreport data source

```
ProductName = "Chicken Delite",
OrderId = "323B61",
Price = 100,
Category = "Non-Veg",
Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
ProductImage = ""

};

datas.Add(data);

data = new ProductList()

{

ProductName = "Chicken Tikka",
OrderId = "323B62",
Price = 64,
Category = "Non-Veg",
Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
ProductImage = ""

};

datas.Add(data);

return datas;
}

}

`
```

Bind the business object data values collection to subreport by adding new item to the **DataSources** as in the following code snippet.

```
`csharp

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //Assigning the data source for 'product-list.rdlc'
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
"list", Value = ProductList.GetData() });
    }
}
```

```
}
```

```
}
```

```
'
```

The data source name is case sensitive, it should be same as in the report definition.

Load subreport stream

To load subreport as stream, set the Stream property in OnInitReportOptions method.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        // Opens the report from application Resources folder using FileStream and loads the sub report stream.
        FileStream reportStream = new
        FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/docs/product-list.rdlc"),
        FileMode.Open, FileAccess.Read);
        reportOption.SubReportModel.Stream = reportStream;
    }
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //Assigning the data source for 'Product List.rdlc'
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
        "list", Value = ProductList.GetData() });
    }
}
```

Report parameters

Provides property options to pass or set report parameters default values at run-time using the [parameters](#) property. You can set the report parameters while creating the Report Viewer control in a script or in the Web API Controller.

In this tutorial, the sales-order-dtail.rdl report is used, and it can be downloaded from [here](#).

Set parameter at client

The [parameters](#) property takes the JSON array value input with parameter details.

Set the default value data to the [values](#) property and name of the report parameter to the [name](#) property.

The parameter name is case sensitive, it should be same as available in the report definition.

The following code example illustrates how to set report parameter in the script.

```
'js
<script type="text/javascript">
$(function () {
  $("#viewer").boldReportViewer({
    reportServiceUrl: "/api/ReportViewer",
    reportPath: '~/Resources/docs/sales-order-detail.rdl',
    parameters: [{ name: 'SalesOrderNumber', labels: ['SO50751'], values: ['SO50751'] }]
  });
})
</script>
'
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location.
For more information, see [Samples and demos](#).

Build and run the application. Preview and edit the result using the following.

```
{% tab demoPath="javascript/report-viewer/report-parameters/set-parameter-at-client"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`

{% endtabItem %}

{% endtab %}
```

Set parameters in Web API Controller

To set parameter default value in Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
```

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    List<BoldReports.Web.ReportParameter> userParameters = new
    List<BoldReports.Web.ReportParameter>();
    userParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = "SalesOrderNumber",
        Values = new List<string>() { "SO50756" }
    });
    reportOption.ReportModel.Parameters = userParameters;
}
```

Get report parameter

The **ReportHelper** class provides methods that help you to get the report parameters used in the report. The following helper methods used to get parameter with or without values.

[Methods](#) | [Description](#)

GetParameters | Returns the parameters that are used in the current report without the processed values.

GetParametersWithValues | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

```
'csharp
```

```
public class ReportsApiController : ApiController, IReportController
{
    Dictionary<string, object> jsonArray = null;
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        jsonArray = jsonResult;
        return ReportHelper.ProcessReport(jsonResult, this);
    }
    ....
```

Report interaction events Report loaded

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    var reportParameters = ReportHelper.GetParameters(jsonArray, this);
    List<BoldReports.Web.ReportParameter> setParameters = new
    List<BoldReports.Web.ReportParameter>();
    if (reportParameters != null)
    {
        foreach (var rptParameter in reportParameters)
        {
            setParameters.Add(new BoldReports.Web.ReportParameter()
            {
                Name = rptParameter.Name,
                Values = new List<string>() { "SO50756" }
            });
        }
        reportOption.ReportModel.Parameters = setParameters;
    }
}
```

Report interaction events

You can handle the report interaction events with reports using the following events.

ReportLoaded

ReportError

ShowError

Drill through

Hyperlink

Report loaded

The [reportLoaded](#) event fires once the report loading is completed and ready to start the processing. You can handle the event and specify data source, parameters at client-side. The following sample code loads a report by assigning report data source input in the [reportLoaded](#) event.

In this tutorial, `product-list.rdlc` report is used, you can add the report from Bold Reports installation location. For more information, see [Samples and demos](#).

```
{% tab demoPath="javascript/report-viewer/report-interaction-events/report-loaded"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

Report error

When an error occurs in the report processing, it raises the [reportError](#) event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```
{% tab demoPath="javascript/report-viewer/report-interaction-events/report-error"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

To suppress the default error dialog, set the cancel argument to true.

Show error

The [showError](#) event invoked whenever the user clicks a report item that contains an error in processing. It provides detailed information about the cause of the error. You can hide the default dialog and show your customized dialog.

```
{% tab demoPath="javascript/report-viewer/report-interaction-events/show-error"
files="index.html,index.js" %}
```

```
{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

Drill through

When a drill through item is selected in a report, it invokes the [drillThrough](#) event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

```
{% tab demoPath="javascript/report-viewer/report-interaction-events/drill-through"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

Hyperlink

The [hyperlink](#) event occurs when the user clicks a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
{% tab demoPath="javascript/report-viewer/report-interaction-events/hyperlink"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
```

```
</body>
`  

{%
  endtabItem %}  

{%
  endtab %}
```

Handle post actions

Report processing actions are sent in an Ajax request to exchange data with the Web API service. You can handle post actions event to customize the Ajax requests.

AjaxBeforeLoad

AjaxSuccess

AjaxError

AjaxBeforeLoad

This event can be triggered before an Ajax request is sent to the Report Viewer Web API service. It allows you to set additional headers, custom data in the Ajax request. The following code sample demonstrates adding custom authorization header and passing default parameter values to service.

Add custom header in Ajax request

Initialize the [ajaxBeforeLoad](#) event in the script and add the authorization token to the `headers` property

```
`js
<script type="text/javascript">
$(function () {
  $("#viewer").boldReportViewer({
    reportServiceUrl: "/api/ReportViewer",
    reportPath: '~/Resources/docs/sales-order-detail.rdl',
    ajaxBeforeLoad: "onAjaxRequest"
  });
  function onAjaxRequest(args) {
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
  }
}</script>
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#).

Get the custom header value from the `HttpContext` header collection using the key name specified at client-side.

`'csharp`

```
string authenticationHeader;  
  
public object PostReportAction(Dictionary<string, object> jsonResult)  
{  
    if (jsonResult != null)  
    {  
        //Get client side custom ajax header and store in local variable  
        authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];  
        //Perform your custom validation here  
        if (authenticationHeader == "")  
        {  
            return new Exception("Authentication failed!!!");  
        }  
        else  
        {  
            return ReportHelper.ProcessReport(jsonResult, this);  
        }  
    }  
    return null;  
}
```

Perform your own action to validate the header values.

Pass custom data in Ajax request

Use the `data` property to set custom data to the server in the Ajax request. In the following code sample, parameter values are passed to the server-side.

`'js`

```
<script type="text/javascript">  
$(function () {  
    $("#viewer").boldReportViewer({  
        reportServiceUrl: "/api/ReportViewer",  
        reportPath: '~/Resources/docs/sales-order-detail.rdl',
```

```
ajaxBeforeLoad: "onAjaxRequest"
});

};

function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}

</script>
`
```

The custom data values are stored in the `customData` header key, you can store it to the local property. The following code sample stores parameter values, then the values are set to report in the `OnReportLoaded` method.

```
`csharp
public string DefaultParameter = null;
string authenticationHeader;
public object PostReportAction(Dictionary<string, object> jsonResult)
{
if (jsonResult != null)
{
if (jsonResult.ContainsKey("customData"))
{
//Get client side custom data and store in local variable. Here parameter values are sent.
DefaultParameter = jsonResult["customData"].ToString();
}

//Get client side custom ajax header and store in local variable
authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
//Perform your custom validation here
if (authenticationHeader == "")
{
return new Exception("Authentication failed!!!");
}
else
{
```

Handle post actions

AjaxSuccess

```
return ReportHelper.ProcessReport(jsonResult, this);
}
}
return null;
}
public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (DefaultParameter != null)
{
//Set client side custom header data
reportOption.ReportModel.Parameters =
JsonConvert.DeserializeObject<List<BoldReports.Web.ReportParameter>>(DefaultParameter);
}
}
`
```

AjaxSuccess

To perform custom action or show user defined message, use the [ajaxSuccess](#) event on the successful Ajax request.

```
'js
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/ReportViewer",
        reportPath: '~/Resources/docs/sales-order-detail.rdl',
        ajaxBeforeLoad: "onAjaxRequest",
        ajaxSuccess: "onAjaxSuccess"
    });
    function onAjaxRequest(args) {
        args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
        //Passing custom parameter data to server
        args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
    }
    function onAjaxSuccess(args) {
```

```
//Perform your custom success message here  
alert("Ajax request success!!!!");  
}  
</script>  
'
```

AjaxError

The [ajaxError](#) event is called, if an error occurred with the request, you can display the customized error detail in the event method.

```
'js  
<script type="text/javascript">  
$(function () {  
    $("#viewer").boldReportViewer({  
        reportServiceUrl: "/api/ReportViewer",  
        reportPath: '~/Resources/docs/sales-order-detail.rdl',  
        ajaxBeforeLoad: "onAjaxRequest",  
        ajaxError: "onAjaxFailure"  
    });  
});  
  
function onAjaxRequest(args) {  
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });  
    //Passing custom parameter data to server  
    args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];  
}  
  
function onAjaxFailure(args) {  
    alert("Status: " + args.status + "\n" +  
        "Error: " + args.responseText);  
}  
</script>  
'
```

You can never have both an error and a success callback with a request.

Error logging in JavaScript Report Viewer

If an error occurred in report processing, JavaScript Report Viewer displays short messages about the error in view. You need to configure the [ApiController](#) to save all logs, stack trace, and error information.

This section explains how to log the detailed error information to your JavaScript application.

This section requires a JavaScript Report Viewer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

In Solution Explorer, open the Report Viewer Controller file.

Inherit the `IReportLogger` interface and implement the interface methods.

```
'csharp
public class ReportViewController : ApiController, IReportController, IReportLogger
{
    public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
    {
        throw new NotImplementedException();
    }

    public void LogError(string errorCode, string message, Exception exception, string errorDetail, string
methodName, string className)
    {
        throw new NotImplementedException();
    }
}
`
```

Create a method in `ReportViewController` to write the error text into application folder.

```
'csharp
internal void WriteLogs(string errorMessage)
{
    string filePath = HttpContext.Current.Server.MapPath("/App_Data/ErrorDetails.txt");
    using (StreamWriter writer = new StreamWriter(filePath, true))
    {
        writer.AutoFlush = true;
        writer.WriteLine(errorMessage);
    }
}
`
```

Invoke the newly created function in `.LogError` as follows.

```
'csharp
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
{
    WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string
methodName, string className)
{
    WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:
{3}", className, methodName, errorDetail, exception.StackTrace));
}
'
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final controller is given as follows, you can replace it in your application.

```
'csharp
public class ReportViewerController : ApiController, IReportController, IReportLogger
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]
    public object GetResource(string key, string resourcetype, bool isPrint)
    {
        return ReportHelper.GetResource(key, resourcetype, isPrint);
    }

    // Method that will be called when initialize the report options before start processing the report
}
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    // You can update report options here
}

// Method that will be called when reported is loaded
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    // You can update report options here
}

public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
{
    WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string
methodName, string className)
{
    WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:
{3}", className, methodName, message, exception.StackTrace));
}

internal void WriteLogs(string errorMessage)
{
    // Error details text file path location
    string filePath = HttpContext.Current.Server.MapPath("/App_Data/ErrorDetails.txt");
    using (StreamWriter writer = new StreamWriter(filePath, true))
    {
        writer.AutoFlush = true;
        writer.WriteLine(errorMessage);
    }
}
`
```

Print report

The Report Viewer provides print report option in the toolbar to print a copy of the report. The page setup dialog allows you to set the paper size or other page setup properties. To see print margins, click **Print Layout** on the toolbar.

The values allow you to set in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values for the page setup dialog come from the report properties, which are set in the design view.

View report in print mode

Print margins are displayed in the Print Layout only, to view report in print mode by default, set the [printMode](#) property to true.

```
{% tab demoPath="javascript/report-viewer/printing/view-report-in-print-mode"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

By default, the Report Viewer renders report in normal layout in which the print margins are not displayed.

Print in new page

To open the print in a new tab of the current browser, set the property [printOption](#) to **NewTab**. By default, it shows the print dialog in the same page.

```
{% tab demoPath="javascript/report-viewer/printing/print-in-new-page" files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`
```

Print report

Set page orientation and paper size

```
{% endtabItem %}  
{% endtab %}
```

The pop-up blocker must be enabled for the page to open the print view in new tab.

Set page orientation and paper size

You can specify the print page paper size, orientation at client-side to change page setup properties by setting the [pageSettings](#) property.

```
{% tab demoPath="javascript/report-viewer/printing/set-page-orientation-and-paper-size"  
files="index.html,index.js" %}  
  
{% tabItem header="index.html" %}  
  
'html  
  
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">  
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>  
<script src="index.js"></script>  
</body>  
'  
  
{% endtabItem %}  
{% endtab %}
```

Set report margin

To set margin values to the report page setup, use the property [margins](#) and specify the value to top, right, bottom, and left.

```
{% tab demoPath="javascript/report-viewer/printing/set-report-margin" files="index.html,index.js" %}  
  
{% tabItem header="index.html" %}  
  
'html  
  
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">  
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>  
<script src="index.js"></script>  
</body>  
'  
  
{% endtabItem %}  
{% endtab %}
```

The values set in the margin property is considered as inches input.

Set page height and width

To set height and width values to the report page setup, use the [height](#), and [width](#) properties.

```
{% tab demoPath="javascript/report-viewer/printing/set-page-height-and-width"  
files="index.html,index.js" %}
```

```
{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

The values set in the height and width property is considered as inches input.

Print report with images

When the report has more images, the browser will send the report stream to the print dialog before the images are completely loaded. To load the print report stream with complete images, you should set the `EmbedImageData` property to true in `OnInitReportOptions` as shown in the following code.

```
`csharp

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.EmbedImageData = true;
}
`
```

Replace the following code sample in client side HTML file.

```
`js

<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer(
    {
        reportServiceUrl: "/api/PrintWithImages",
        reportPath: '~/Resources/docs/product-details.rdl'
    });
});
</script>
`
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

In this tutorial, the `product-details.rdl` report is used, and it can be downloaded from [here](#).

External styles in report printing

While printing report, the external styles are used in the application overrides printable page style and prints output with incorrect alignments. To avoid the external script overriding, set the `isStyleLoad` property to false, which will print the page using only the Report Viewer styles.

```
'js
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer(
    {
        reportServiceUrl: "/api/ReportViewer",
        reportPath: '~/Resources/docs/product-details.rdl',
        reportPrint: "onReportPrint"
    });
});
function onReportPrint(args) {
    args.isStyleLoad = false;
}
</script>
'
```

Show print progress

Report Viewer provides events that help you to show the progress information when the printing takes a long time to complete.

Set the [printProgressChanged](#) in Report Viewer initialization.

Implement the function and add code samples to show a custom message based on the print progress status as shown in the following code snippet.

```
{% tab demoPath="javascript/report-viewer/printing/show-print-progress" files="index.html,index.js"
%}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
```

Export report

Remove empty spaces in printing

```
`  
{%- endtabItem %}  
{%- endtab %}
```

Remove empty spaces in printing

The extra blank page is created when the Body of your report is too wide for your page. If you want the report to appear on a single page, all the content within the report body must fit on the physical page and the body width should be lesser or equal to the following formula:

Body Width <= Page Width - (Left Margin + Right Margin)

For more details on designing a report to remove the empty pages in the report, refer to the knowledge base article of [report page sizing](#).

Export report

The Report Viewer provides events and properties to control and customize the report export functionality.

Export event handling

You can show the progress information, when the exporting takes long time to complete using the [exportProgressChanged](#) event.

Set the [exportProgressChanged](#) in Report Viewer initialization.

Implement the function and replace the following code samples to show a custom message based on the progress stage.

```
{%- tab demoPath="javascript/report-viewer/export/export-event-handling" files="index.html,index.js" %}  
{%- tabItem header="index.html" %}  
`html  
  
  
{%- endtabItem %}  
{%- endtab %}
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location.
For more information, see [Samples and demos](#).

Export data visualization items

To export the reports with data visualization components, it is mandatory to configure the web scripts in Report Viewer Web API controller. If the report definition uses chart, gauge and map report items then configure the scripts in Web API as the following steps,

Open the Report Viewer Web API controller.

Configure the below scripts and styles in **OnInitReportOptions** on Web API controller.

jquery-1.10.2.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js

ej2-base.min.js

ej2-data.min.js

ej2-pdf-export.min.js

ej2-svg-base.min.js

ej2-lineargauge.min.js

ej2-circulargauge.min.js

ej.map.min.js

bold.report-viewer.min.js

You can replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
    }
}
```

```
//Chart component script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
//Gauge component scripts
resourcesPath + @"\bold-reports\common\ej2-base.min.js",
resourcesPath + @"\bold-reports\common\ej2-data.min.js",
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
//Report Viewer Script
resourcesPath + @"\bold-reports\bold.report-viewer.min.js",
};

reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
    resourcesPath + @"jquery-1.10.2.min.js"
};

}
`
```

The data visualization components will not export without the above script configurations.

[Export data visualization items in azure environment](#)

Report Viewer uses **WebBrowser** to export the data visualization items to PDF, Word, Excel file formats. The **WebBrowser** is not supported in azure environment. To overcome this limitation in azure environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

To download **PhantomJS** application and deploy it on your machine, you should accept it's license terms on [LICENSE](#) and [Third-Party](#) document.

Download **PhantomJS** from [here](#) and extract the download file.

Copy the **PhantomJS.exe** file from the extracted bin folder and paste into **PhantomJS** folder in your application.

Open the Report Viewer Web API controller.

Set the `UsePhantomJS` property to true and `PhantomJSPPath` property in `OnInitReportOptions` method.

```
'csharp
// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ExportResources.UsePhantomJS = true;
    reportOption.ReportModel.ExportResources.PhantomJSPPath = @"\PhantomJS\";
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
        //Gauge component scripts
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
        //Map component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
        //Report Viewer Script
        "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
    };
    reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
    {
        "https://code.jquery.com/jquery-1.10.2.min.js"
    };
}
`
```

The **Scripts** and **Dependent** scripts must be added to export the items. For more details refer to the [export data visualization items](#) section.

Change Excel and Word export format

This allows you to change the default file format to any other file format using the [excelFormat](#) and [wordFormat](#) properties. The following code sample changes the default versions.

```
{% tab demoPath="javascript/report-viewer/export/change-excel-and-word-export-format"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

Hide specific export type for report

Show or hide the default export types available in the component using the [exportOptions](#) property. The following code hides the HTML export type from the default export options.

```
{% tab demoPath="javascript/report-viewer/export/hide-specific-export-type-for-report"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

PDF export options

The **PDFOptions** provides properties to manage PDF export behaviors. You have to set the properties in the **OnInitReportOptions** method of the Web API service.

Export with complex scripts

To export reports with the complex scripts, set the **ComplexScript** property of **PDFOptions** instance to true.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
    {
        EnableComplexScript = true
    };
}
`
```

PDF Conformance

You can export the report as PDF/A-1b document by specifying the conformance level PdfConformanceLevel.Pdf_A1B in the PdfConformanceLevel property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
    {
        PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B
    };
}
`
```

Add custom PDF fonts

This allows you to have custom fonts in the PDF exported document by adding the font streams to Fonts collection in PDFOptions instance.

Add the font .ttf files into your application Resources folder.

In the Solution Explorer, open the properties of the font file and set the property Copy to Output Directory as Copy always.

Initialize the Font collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
```

```
{  
//Load Missing font stream  
Fonts = new Dictionary<string, System.IO.Stream>  
{  
{ "Segoe UI",  
System.IO.File.OpenRead(System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/docs/fon  
t_symbols.ttf")) }  
}  
};  
}  
`
```

Any fonts used in the report definition that is not installed or available in the local system, then you must load the font stream. In the above code, loaded `font_symbols` font stream to export `Sales Order Detail.rdl` report as symbols.

Word export options

The `WordOptions` provides properties to manage Word document export behaviors.

Word document type

You can save the report to the required document version by setting the `FormatType` property.

```
`csharp  
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
{  
FormatType = BoldReports.Writer.WordFormatType.Docx  
};  
`
```

Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the `LayoutOption` as `TopLevel`. The `ParagraphSpacing` is the distance value added between two elements in the document.

```
`csharp  
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
{  
LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel,  
ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()  
{  
Bottom = 0.5f,
```

```
Top = 0.5f  
}  
};  
,
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in word document is not a stand-alone object, if you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, added an empty paragraph between two tables.

Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without a password. The following are the types of protection.

AllowOnlyComments: You can add or modify only the comments in the Word document.

AllowOnlyFormFields: You can modify the form field values in the Word document.

AllowOnlyRevisions: You can accept or reject the revisions in the Word document.

AllowOnlyReading: You can only view the content in the Word document.

NoProtection: You can access or edit the Word document contents as normally.

```
`csharp  
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
{  
    ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading  
};  
,
```

Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

```
`csharp  
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013  
};
```

Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the `LayoutOption` as `IgnoreCellMerge`.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
    LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge
};
```

Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the `ExcelSheetProtection` or enabling the `ReadOnlyRecommended` properties.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
    ReadOnlyRecommended = true,
    ExcelSheetProtection = ExcelSheetProtection.DeletingColumns
};
```

PowerPoint export options

You can save the report to the required PowerPoint version by setting the `FormatType` property.

```
'csharp
```

```
reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()
{
    FormatType = BoldReports.Writer.PPTSaveType.PowerPoint2013
};
```

CSV export options

The `CsvOptions` allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
'csharp
```

```
reportOption.ReportModel.CsvOptions = new BoldReports.Writer.CsvOptions()
{
    Encoding = System.Text.Encoding.Default,
```

```
FieldDelimiter = ",",
UseFormattedValues = false,
Qualifier = "#",
RecordDelimiter = "@",
SuppressLineBreaks = true,
FileExtension = ".txt"
};

`
```

HTML export options

You can hide the separator added at the end of each page by setting the `HidePageSeparator` property to true.

```
'csharp
reportOption.ReportModel.HTMLOptions = new BoldReports.Writer.HTMLOptions()
{
    HidePageSeparator = true
};
```

Password protect exported document

This allows you to protect the exported document such as PDF, Word, Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet illustrates how to encrypt the exported document with the user defined password.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //PDF encryption
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions();
    reportOption.ReportModel.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
    {
        UserPassword = "password"
    };
    //Word encryption
    reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
    {
        EncryptionPassword = "password"
    };
}
```

```
};

//Excel encryption
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
    PasswordToModify = "password",
    PasswordToOpen = "password"
};
//PPT encryption
reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()
{
    EncryptionPassword = "password"
};
}

`
```

Password protection is not supported for HTML export format.

Toolbar customization

You can hide the component toolbar to show customized user interface or to customize the toolbar icons and elements appearances using the templates and Report Viewer toolbar customization properties.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#). You can add the reports from Syncfusion installation location. For more information, see [Samples and demos](#).

Hide parameter block and toolbar items

To hide toolbar items, set the `toolbarSettings` property. The following code can be used to remove the parameter option from the toolbar and hide the parameter block.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/hide-parameter-block"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}
```

```
{% endtab %}
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location. For more information, see [Samples and demos](#).

The following code sample hides the print options from the toolbar items.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/hide-toolbar-items"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

Similarly, you can show or hide all other toolbar options with the help of [toolbarSettings.items](#) enum.

Enable stop option in toolbar

To enable stop option in toolbar, set the [toolbarSettings.items](#) property to `ej.ReportViewer.ToolbarItems.All`. The following code can be used to enable stop option in toolbar.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/enable-stop-option-in-toolbar"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}

{% endtab %}
```

Hide toolbar

To hide the Report Viewer toolbar set the [showToolbar](#) property to false.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/hide-toolbar"
files="index.html,index.js" %}

{% tabItem header="index.html" %}
```

```
'html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
'

{%- endtabItem %}

{%- endtab %}
```

Decide or hide the export option

The Report Viewer provides the [exportOptions](#) property to show or hide the default export types available in the component. The following code hides the HTML export type from the default export options.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/decide-or-hide-the-export-option"
files="index.html,index.js" %}

{%- tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
'

{%- endtabItem %}

{%- endtab %}
```

Add custom items to the export drop-down

To add custom items to the export drop-down available in the Report Viewer toolbar, use the property [customItems](#) and provide the JSON array of collection input with the [index](#), [cssClass](#) name, and [value](#) properties. Register the [exportItemClick](#) event to handle the custom item actions as given in following code snippet..

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/add-custom-items-to-the-export-
drop-down" files="index.html,index.js" %}

{%- tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
```

```
`  
{%- endtabItem %}  
{%- endtab %}
```

Add custom toolbar item

You can add custom items to Report Viewer toolbar using the [toolbarSettings](#) property. You must register the [toolBarItemClick](#) event to handle the newly added custom items action.

Add custom item to exiting toolbar group

To add a custom item to existing toolbar group use the property [customItems](#) in [toolbarSettings](#) and provide the JSON array of collection input with the [groupIndex](#), [index](#), [itemType](#), [cssClass](#) name, and [tooltip](#) properties as given in following code snippet.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/add-custom-toolbar-item/add-  
custom-item-to-existing-toolbar-group" files="index.html,index.js" %}
```

```
{%- tabItem header="index.html" %}
```

```
`html
```

```
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">  
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>  
<script src="index.js"></script>  
</body>
```

```
`  
{%- endtabItem %}
```

```
{%- endtab %}
```

Add new toolbar group

To add new toolbar group and custom items to it, use the property [customGroups](#) in [toolbarSettings](#) and provide the JSON array of collection input with the [groupIndex](#), [items](#) properties. The [items](#) must have the properties [itemType](#), [cssClass](#) and [tooltip](#) as given in following code snippet.

```
{% tab demoPath="javascript/report-viewer/toolbar-customization/add-custom-toolbar-item/add-new-  
toolbar-group" files="index.html,index.js" %}
```

```
{%- tabItem header="index.html" %}
```

```
`html
```

```
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">  
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>  
<script src="index.js"></script>  
</body>
```

```
`  
{%- endtabItem %}
```

```
{%- endtab %}
```

Custom Actions

This section explains you the steps required to add user defined buttons in Report Viewer toolbar and invoke custom actions.

Send a report as an email attachment

This topic describes steps required to create custom email option and share a report as an email attachment to other users.

Add email button in Report Viewer

Create email button option in the toolbar using the [customItems](#) property with the values such as `groupIndex, index, itemType, cssClass, tooltip, toolBarItemClick` event to fire when you click the button.

Access the Report Viewer model and create a JSON array for sending requests to the Web API server. You can use the following codes for creating the event with custom action.

```
'js
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/ReportViewer",
        reportPath: '~/Resources/docs/sales-order-detail.rdl',
        toolbarSettings: {
            showToolbar: true,
            items: ej.ReportViewer.ToolbarItems.All & ~ej.ReportViewer.ToolbarItems.Print,
            customItems: [{
                groupIndex: 1,
                index: 1,
                type: 'Default',
                cssClass: "e-icon e-mail e-reportviewer-icon",
                id: 'E-Mail',
                tooltip: {
                    header: 'E-Mail',
                    content: 'Send rendered report as mail attachment'
                }
            }]
        },
        toolBarItemClick: 'ontoolBarItemClick'
    })
})'
```

```
});
});

//Toolbar click event handler
function onToolBarItemClick(args) {
if (args.value == "E-Mail") {
var proxy = $('#viewer').data('boldReportViewer');
var Report = proxy.model.reportPath;
var lastIndex = Report.lastIndexOf("/");
var reportName = Report.substring(lastIndex + 1);
var requrl = proxy.model.reportServiceUrl + '/SendEmail';
var _json = {
exportType: "PDF", reportViewerToken: proxy._reportViewerToken, ReportName: reportName
};
$.ajax({
type: "POST",
contentType: "application/json; charset=utf-8",
url: requrl,
data: JSON.stringify(_json),
dataType: "json",
crossDomain: true
})
}
}

</script>
`
```

You can view the Web API service used in the above code from the [Reporting Service](#) git hub location.
For more information, see [Samples and demos](#).

Create custom email action

Create a new action method `SendEmail` in the Web API service.

Export the report to the required type using `ReportHelper.GetReport` to send report stream as an attachment.

The following code sample exports the report to stream and send it as an attachment to a specified mail address. In the code, `SmtpClient` is used to send the report as an email attachment.

```
`csharp
public object SendEmail(Dictionary<string, object> jsonResult)
{
    string _token = jsonResult["reportViewerToken"].ToString();
    var stream = ReportHelper.GetReport(_token, jsonResult["exportType"].ToString());
    stream.Position = 0;
    if (!ComposeEmail(stream, jsonResult["reportName"].ToString()))
    {
        return "Mail not sent !!!";
    }
    return "Mail Sent !!!";
}

public bool ComposeEmail(Stream stream, string reportName)
{
    try
    {
        MailMessage mail = new MailMessage();
        SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");
        mail.IsBodyHtml = true;
        mail.From = new MailAddress("xx@gmail.com");
        mail.To.Add("xx@gmail.com");
        mail.Subject = "Report Name : " + reportName;
        stream.Position = 0;
        if (stream != null)
        {
            ContentType ct = new ContentType();
            ct.Name = reportName + DateTime.Now.ToString() + ".pdf";
            System.Net.Mail.Attachment attachment = new System.Net.Mail.Attachment(stream, ct);
            mail.Attachments.Add(attachment);
        }
        SmtpServer.Port = 587;
        SmtpServer.Credentials = new System.Net.NetworkCredential("xx@gmail.com", "xx");
        SmtpServer.EnableSsl = true;
    }
}
```

```
SmtpServer.Send(mail);
return true;
}
catch (Exception ex)
{
return ex.ToString();
}
return false;
}
`
```

In the above code sample, the report is exported to PDF and sent to users using `SmptClient`.

Build and run the application. Preview and edit the result using the following.

```
{% tab demoPath="javascript/report-viewer/custom-actions/add-email-button"
files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}
{% endtab %}
```

Cancel report processing in Report Viewer

This topic describes steps required to create custom cancel option to cancel the report processing in Report Viewer.

Add cancel report button

Create a cancel button in the toolbar using the `customItems` property with the values such as `cssClass`, `tooltip`.

Register the event `toolBarItemClick`, `ajaxSuccess`, `ajaxBeforeLoad`, `renderingComplete` to handle the cancel button click action.

You can use the following client side codes to create the cancel button.

```
'js
<script type="text/javascript">
$(function () {
    $("#viewer").boldReportViewer({
        reportServiceUrl: "/api/ReportViewer",
        reportPath: '~/Resources/docs/product-line-sales.rdl',
        toolbarSettings: {
            showToolbar: true,
            customItems: [{
                type: 'Default',
                cssClass: "e-icon e-close e-reportviewer-icon",
                id: 'Stop',
                tooltip: {
                    header: 'Stop',
                    content: 'Stop processing the report'
                }
            }]
        },
        toolBarItemClick: 'onToolBarItemClick',
        ajaxSuccess: "onAjaxSuccess",
        ajaxBeforeLoad: "onAjaxRequest",
        renderingComplete: "onRenderingComplete"
    });
});
//Toolbar click event handler
function onToolBarItemClick(args) {
    if (args.value === "Stop") {
        var proxy = $('#viewer').data('boldReportViewer');
        proxy.cancelRendering();
        $('#viewertoolbarStop').addClass('e-disable');
    }
}
//Ajax request event handler to enable cancel button
```

```

function onAjaxRequest() {
    $('#viewertoolbarStop').removeClass('e-disable');
}
//Ajax success event handler to disable cancel button
function onAjaxSuccess() {
    $('#viewertoolbarStop').addClass('e-disable');
}
//Rendering complete event handler to disable cancel button
function onRenderingComplete() {
    $('#viewertoolbarStop').addClass('e-disable');
}

```

</script>

,

Build and run the application. Preview and edit the result using the following.

```

{%
    tab demoPath="javascript/report-viewer/custom-actions/add-cancel-report-button"
    files="index.html,index.js"
}

{%
    tabItem header="index.html"
}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
    <div id="viewer" style="position: absolute; height: 100%; width: 100%;"></div>
    <script src="index.js"></script>
</body>

`


{%
    endtabItem
}

{%
    endtab
}

```

Localization of Bold Reports HTML5 JavaScript Report Viewer

Localization of HTML5 JavaScript Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. To render the static text with specific culture, refer to the following corresponding culture script files and set culture name to the [locale](#) property of the Report Viewer.

[ej.localetexts.fr-FR.min.js](#)

[ej.culture.fr-FR.min.js](#)

Refer the ej.localetexts.fr-FR.min.js from cdn using the following code.

```
'html
```

```
<script src="http://cdn.boldreports.com/2.2.32/scripts/l10n/ej.localetexts.fr-FR.min.js"></script>
```

```
'
```

Refer the ej.culture.fr-FR.min.js from cdn using the following code.

```
'html
```

```
<script src="http://cdn.boldreports.com/2.2.32/scripts/i18n/ej.culture.fr-FR.min.js"></script>
```

```
'
```

Whether you want to get the localization script as local then install the BoldReports.Global NuGet package in your application.

```
'html
```

```
<script src="scripts/l10n/ej.localetexts.fr-FR.min.js"></script>
```

```
<script src="scripts/i18n/ej.culture.fr-FR.min.js"></script>
```

```
'
```

You can edit and preview the Report Viewer localization using the following.

```
{% tab demoPath="javascript/report-viewer/localization" files="index.html,index.js" %}
```

```
{% endtab %}
```

Responsive layout rendering of HTML5 JavaScript Report Viewer

Report Viewer will adaptively render itself with optimal user interfaces for phone, tablet, or desktop form factors. This helps your application to scale elegantly on all form factors with ease. You can enable responsive layout rendering in Report Viewer by setting [isResponsive](#) property to true.

```
'html
```

```
<div style="height: 600px; width: 950px;">  
  <!-- Creating a div tag which will act as a container for boldReportViewer widget.-->  
  <div style="height: 600px; width: 950px; min-height: 400px;" id="viewer"></div>  
  <!-- Setting property and initializing boldReportViewer widget.-->  
  <script type="text/javascript">  
    $(function () {  
      $("#viewer").boldReportViewer({  
        reportServiceUrl: "https://demos.boldreports.com/services/api/ReportViewer",  
        reportPath: '~/Resources/docs/sales-order-detail.rdl',
```

| | |
|-------------|---------------|
| Limitations | Normal layout |
|-------------|---------------|

```

isResponsive: true
});
});
</script>
</div>
`
```

Normal layout

The following output shows the normal layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in normal layout](/static/assets/javascript/report-viewer/images/responsive-layout/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/javascript/report-viewer/images/responsive-layout/responsive-layout.png)

Limitations

RDL Specification

The Report Viewer control does not support RDL Specification for SQL Server 2000 and SQL Server 2005.

Report Layout

Vertical alignment in the text box report item is not supported in web rendering.

In the Tablix cell split layout process when the table cell width value exceeds the page width, the entire cell moves to the next page to display the complete cell items.

Expressions

The object function and VB function do not have complete support.

SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you must specify the data source credentials for each report data source to establish database connection.

Samples and demos

Browse and explore our ready-to-use RDL, RDLC reports, samples, online and offline demos.

Locally installed reports

You can obtain sample rdl and rdlc files from Bold Reports installed location

%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view the JavaScript Report Viewer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view the GitHub Report Viewer demo samples.

Reporting Service application

Click [here](#) to view the GitHub source location of the reporting services that are used in our documentation.

Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of JavaScript Report Viewer application:

Client-side migration

To refer the latest scripts and CSS references of Report Viewer do the following steps,

Remove the following scripts and CSS references from the Report Viewer page.

`ej.web.all.min.css`

`ej.web.all.min.js`

Replace the following scripts and CSS references in the <head> tag of the Report Viewer page.

```
'html
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="http://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
'
```

Adding data visualization scripts

To render the report with data visualization components such as chart, gauge, and map items, add scripts of the visualization element. The following table shows the script reference that needs to be added in Report Viewer page for data visualization elements.

Visualization item | Script file

Chart | ej.chart.min.js|

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item, add chart control script ej.chart.min.js before the bold.report-viewer.min.js reference in the \Views\Shared_Layout.cshtml page as demonstrated in the following code sample.

```
'html
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!-Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

The following code can be used to render the chart, gauge, and map report items in Report Viewer.

```
'html
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!-Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="Scripts/bold-reports/common/ej2-base.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-data.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-pdf-export.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-svg-base.min.js"></script>
```

```
<script src="Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js"></script>
<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

Server-side migration

In the Solution Explorer, right-click the **References** and remove the **Syncfusion.EJ.ReportViewer** assembly reference.

Add the assembly from the Bold Reporting NuGet package **BoldReports.Web**. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for **BoldReports.Web** NuGet package, and then install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Web API Controller

The **IReportController** interface is moved to **BoldReports.Web.ReportViewer**. Open the Report Viewer Web API Controller file and remove the following using statement.

```
`csharp
using Syncfusion.EJ.ReportViewer;
`
```

Add the following using statement.

```
`csharp
using BoldReports.Web.ReportViewer;
`
```

Your application is successfully upgraded to the latest version of Report Viewer, and you can run the application with new assemblies.

Report export configuration

Now, the **BoldReports.Web** can export the reports with data visualization components only using web components. It is mandatory to configure the web scripts in Report Viewer Web API controller for exporting data visualization components such as chart, gauge, and map that are used in report definition. To configure the scripts in Web API, refer to the following steps:

Open the Report Viewer Web API controller.

Configure the following scripts and styles in `OnInitReportOptions` on Web API controller:

`jquery-1.10.2.min.js`

`bold.reports.common.min.js`

`bold.reports.widgets.min.js`

`ej.chart.min.js` - Exports the chart item. Add this script only if your report contains the chart report item.

`ej2-base.min.js`, `ej2-data.min.js`, `ej2-pdf-export.min.js`, `ej2-svg-base.min.js`, `ej2-lineargauge.min.js` and `ej2-circulargauge.min.js` - Exports the gauge item. Add this script only if your report contains the gauge report item.

`ej.map.min.js` - Exports the map item. Add this script only if your report contains the map report item.

`bold.report-viewer.min.js`

Replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        //Gauge component scripts
        resourcesPath + @"\bold-reports\common\ej2-base.min.js",
        resourcesPath + @"\bold-reports\common\ej2-data.min.js",
        resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
        resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
        resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
    }
}
```

```
resourcesPath + @"\\bold-reports\\data-visualization\\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.map.min.js",
//Report Viewer Script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.chart.min.js",
resourcesPath + @"\\bold-reports\\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\\jquery-1.7.1.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

NuGet Packages for JavaScript

Refer to the following steps to configure Bold Reports NuGet packages for JavaScript application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: [NuGet.org](#)

Source: <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/javascript/report-viewer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reporting NuGet packages are shipped into our Bold Reporting Tools build. To configure the packages from Bold Reports installed location, use the following steps:

install specified package in default project

Installing NuGet packages

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages.

![Offline NuGet Configure](/static/assets/javascript/report-viewer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

Installing NuGet packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**.

Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages **BoldReports.Web**, and then click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

install specified package in default project

```
Install-Package <Package Name>
```

install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

For example:`cmd`**install specified package in default project**`Install-Package BoldReports.Web`**install specified package in default project with specified Package Source**`Install-Package BoldReports.Web -Source "https://api.nuget.org/v3/index.json"`**install specified package in specified project**`Install-Package BoldReports.Web -ProjectName BoldReportsApplication`**Upgrading NuGet packages****Upgrading using NuGet Package Manager**

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**

Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources,

select the required packages and specific version from the drop-down, and then click the **Update** button.

Upgrading using Package Manager Console

To update the installed Bold Reporting NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

`cmd`**Update specific NuGet package in default project**`Update-Package <Package Name>`**Update all the packages in default project**`Update-Package`

Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

\

For example:

`cmd

Update specified Bold Reporting NuGet package

Update-Package BoldReports.Web

Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportsApplication

\

Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing `nuget.exe` to latest version use the following command:

`cmd

nuget update -self

\

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting NuGet packages.

`cmd

update all NuGet packages from config file

nuget update <configPath> [options]

update all NuGet packages from specified Packages Source

nuget update -Source <Source Location> [optional]

\ configPath is optional. It identifies the package.config or solutions file lists the packages utilized in the project.

For example:

`cmd

Update all NuGet packages from config file

nuget update "C:\Users\BoldReportsApplication\package.config"

Update all NuGet packages from specified Packages Source

nuget update -Source "https://api.nuget.org/v3/index.json"

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

CDN

The [CDN](#) links are provided individually for all the scripts and style sheets of Bold Reports JavaScript Reporting component.

CDN scripts links

Bold Reports dependency libraries

The CDN script files are maintained for each version of the Report Platform SDK individually.

The three scripts, namely bold.reports.common.min.js, bold.reports.widgets.min.js, and bold.report-viewer.min.js are mandatory to render the Bold Report Viewer.

| Name | Details | CDN link |
|-----------------------------|------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| bold.reports.common.min.js | Common script for reporting widgets. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js |
| bold.reports.widgets.min.js | Supports Syncfusion widgets to render in | Secured link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js |

| | | |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | HTML5 format. | |
| ej.chart.min.js | Renders the chart item. Add this script only if your report contains the chart report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js |
| ej2-base.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js |
| ej2-data.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js |

| | | |
|------------------------|----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej2-pdf-export.min.js | <p>Renders the gauge item. Add this script only if your report contains the gauge report item.</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js Unsecured Link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js</p> |
| ej2-svg-base.min.js | <p>Renders the gauge item. Add this script only if your report contains the gauge report item.</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js Unsecured Link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js</p> |
| ej2-lineargauge.min.js | <p>Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js Unsecured link:http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js</p> |

| | | |
|---------------------------|------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | |
| ej2-circulargauge.min.js | Renders the circular gauge item. Add this script only if your report contains the circular gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js |
| ej.map.min.js | Renders the map item. Add this script only if your report contains the map report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js |
| bold.report-viewer.min.js | Renders the Syncfusion JavaScript Report Viewer widget. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js |

External dependency libraries

The basic syntax is,

`https://cdn.boldreports.com/external/fileName`

Example: `https://cdn.boldreports.com/external/jquery-1.10.2.min.js`

| Name | Details | CDN link |
|------|---------|----------|
| | | |

| | | |
|------------------|----------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| jQuery 1.10.2 | Common jQuery script to render the Syncfusion JavaScript Reporting widgets | Secured link: https://cdn.boldreports.com/external/jquery-1.10.2.min.js Unsecured link: http://cdn.boldreports.com/external/jquery-1.10.2.min.js |
|------------------|----------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

CDN style sheet links

The CDN links for all the CSS files are provided in the following table. Refer to the following syntax:

[https://cdn.boldreports.com/\[version\]/content/\[theme-name\]/\[fileName\]](https://cdn.boldreports.com/[version]/content/[theme-name]/[fileName])

| Name | Details | CDN link |
|-----------------------------|---------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Material (default theme) | Includes the CSS properties defined for the JavaScript Reporting component in material. (Default-theme) | Secured link: https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css Unsecured link: http://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css |
| Office-365 | Includes the CSS properties defined for the JavaScript Reporting component in office-365 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/office-365/bold.reports.all.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/office-365/bold.reports.all.min.css |
| High-contrast-01 | Includes the CSS properties defined for the JavaScript Reporting component in high- | Secured Link: https://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reports.all.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reports.all.min.css |

| | | |
|------------------|-------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | contrast-01 theme. | |
| High-contrast-02 | Includes the CSS properties defined for the JavaScript Reporting component in high-contrast-02 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reports.all.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reports.all.min.css |
| Bootstrap-theme | Includes the CSS properties defined for the JavaScript Reporting component in bootstrap theme. | Secured link: https://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reports.all.min.css Unsecured link: http://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reports.all.min.css |

All the provided CDN links can be accessed either through `http` or `https`.

Refer local Scripts and CSS when CDN fails

One of the major risks with CDN links is, sometimes, it may go down due to the network or connection problems. On such scenarios, you can refer the local scripts and CSS files dynamically in application by checking if the scripts and CSS files are loaded through CDN that returns `undefined` as demonstrated in the following code sample.

```
'html
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>My first HTML page</title>
// CDN LINK references
<link
href="http://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="http://cdn.boldreports.com/external/jquery-1.10.2.min.js" type="text/javascript"></script>
```

```
<script
src="http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
<script type="text/javascript">
if (typeof jQuery == "undefined") { // If CDN fails, jQuery returns undefined
// Referring local scripts - Specify the path where the below files are located in your machine
document.write(decodeURIComponent("%3Cscript src="Scripts/jquery-1.10.2.min.js"
%3E%3C/script%3E"));
}
if (typeof ej == "undefined") { // If CDN fails, ej returns undefined.
// Refer the Syncfusion stylesheets and scripts from the local path here
// StyleSheet reference from the local system path
document.write(decodeURIComponent("%3Clink rel="stylesheet" href="Content/bold-
reports/material/bold.reports.all.min.css" %3C/%3E"));
document.write(decodeURIComponent("%3Cscript src="Scripts/bold-
reports/common/bold.reports.common.min.js" %3E%3C/script%3E"));
document.write(decodeURIComponent("%3Cscript src="Scripts/bold-
reports/common/bold.reports.widgets.min.js" %3E%3C/script%3E"));
// Script reference from the local system path
document.write(decodeURIComponent("%3Cscript src="Scripts/bold-reports/bold.report-viewer.min.js"
%3E%3C/script%3E"));
}
</script>
</head>
<body>
<script type="text/javascript">
$(function () {
// initialization boldReportViewer
$("#viewer").boldReportViewer();
});
</script>
</body>
</html>
`
```

JavaScript Report Viewer Reporting Service

The JavaScript Report Viewer requires a Web API service to process the report files. The following topics explains how to create reporting Web API service to preview the reports.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

IReportController

The interface `IReportController` has the declaration of action methods that is defined in Web API Controller for processing the RDL, RDLC, SSRS report and handling request from Report Viewer control. The `IReportController` has the following action methods declaration.

[Methods](#) | [Description](#)

`GetResource` | Returns the report resource for the requested key.

`ProcessReport` | Processes the report request and returns the result.

'csharp

```
public class ReportsController: ApiController,IReportController
{
    /// <summary>
    /// Action (HttpGet) method for getting resource for report.
    /// </summary>
    /// <param name="key">The unique key to get the required resource.</param>
    /// <param name="resourceType">The type of the requested resource.</param>
    /// <param name="isPrinting">If set to <see langword="true"/>, then the resource is generated for printing.</param>
    /// <returns>The object data.</returns>
    public object GetResource(string key, string resourcetype, bool isPrint)
    {
        //Returns the report resource for the requested key.
        return ReportHelper.GetResource(key, resourcetype, isPrint);
    }
    /// <summary>
    /// Report Initialization method that is triggered when report begin processed.
    /// </summary>
    /// <param name="reportOptions">The ReportViewer options.</param>
    public void OnInitReportOptions(ReportViewerOptions reportOptions)
```

```
{  
//You can update report options here  
}  
/// <summary>  
/// Report loaded method that is triggered when report and sub report begins to be loaded.  
/// </summary>  
/// <param name="reportOptions">The ReportViewer options.</param>  
public void OnReportLoaded(ReportViewerOptions reportOptions)  
{  
//You can update report options here  
}  
/// <summary>  
/// Action (HttpPost) method for posting the request for report process.  
/// </summary>  
/// <param name="jsonData">The JSON data posted for processing report.</param>  
/// <returns>The object data.</returns>  
public object PostReportAction(Dictionary < string, object > jsonData)  
{  
//Processes the report request and returns the result.  
return ReportHelper.ProcessReport(jsonData, this);  
}  
}  
`
```

Create ASP.NET Web API service

In this section, you will learn how to create a Web API Service for Report Viewer using the new ASP.NET Empty Web Application template.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![ASP.NET Web Application project template](/static/assets/javascript/report-viewer/images/report-service/aspnetmvc5-template.png)

Choose **Empty, Web API** and then click **OK**. Now, the Web application project is created with default ASP.NET Web template.

![Select Web API and Empty options](/static/assets/javascript/report-viewer/images/report-service/add-web-api.png)

Configure Report Viewer Web API

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web** NuGet packages, and install them in your Web application.

Package | Purpose

PostReportAction | Action (HttpPost) method for posting the request in report process.

OnInitReportOptions | Report initialization method that occurs when the report is about to be processed.

OnReportLoaded | Report loaded method that occurs when the report and sub report start loading.

GetResource | Action (HttpGet) method to get resource for the report.

ReportHelper

The class **ReportHelper** contains helper methods that help to process a Post or Get request from the Report Viewer control and return the response to the Report Viewer control. It has the following methods:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

Right-click **Controller** folder in your project and select **Add > New Item** from the context menu.

Select **Web API Controller Class** from the listed templates and name it as
ReportViewController.cs

![Provide controller name](/static/assets/javascript/report-viewer/images/report-service/add-web-api-controller.png)

Click **Add**.

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
public class ReportViewerController : ApiController, IReportController
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]
    public object GetResource(string key, string resourcetype, bool isPrint)
    {
        return ReportHelper.GetResource(key, resourcetype, isPrint);
    }

    // Method that will be called when initialize the report options before start processing the report
    public void OnInitReportOptions(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }

    // Method that will be called when reported is loaded
    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }
}
```

Add routing information

To configure routing to include an action name in the URI, open the `WebApiConfig.cs` file and change the `routeTemplate` in the `Register` method as follows,

```
'csharp
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Web API configuration and services
        // Web API routes
        config.MapHttpAttributeRoutes();

        config.Routes.MapHttpRoute(
            name: "DefaultApi",
            routeTemplate: "api/{controller}/{action}/{id}",
            defaults: new { id = RouteParameter.Optional }
        );
    }
}
```

Compile and run the Web API service application.

Enable Cross-Origin Requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for `Microsoft.AspNet.WebApi.Cors` NuGet packages, and install them in your Web API application.

Call `EnableCors` in `WebApiConfig` to add CORS services to the `Register` method. Replace the following code to allow any origin requests.

```
`csharp
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Add Enable Cors
        config.EnableCors();

        // Web API configuration and services
        // Web API routes
        config.MapHttpAttributeRoutes();

        config.Routes.MapHttpRoute(
            name: "DefaultApi",
            routeTemplate: "api/{controller}/{action}/{id}",
            defaults: new { id = RouteParameter.Optional }
        );
    }
}
`
```

To specify the CORS policy for API controller, add the `[EnableCors]` attribute to the controller class. Specify the policy name.

```
`csharp
[EnableCors(origins: "", headers: "", methods: "*")]
public class ReportViewerController : ApiController, IReportController
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]

    public object GetResource(string key, string resourcetype, bool isPrint)
```

```
{  
    return ReportHelper.GetResource(key, resourcetype, isPrint);  
}  
  
// Method that will be called when initialize the report options before start processing the report  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    // You can update report options here  
}  
  
// Method that will be called when reported is loaded  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    // You can update report options here  
}  
}
```

Create ASP.NET Core Web API Service

In this section, you will learn how to create a ASP.NET Core Web API for Report Viewer using the new ASP.NET Core Web Application template.

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Viewer API creation.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/javascript/report-viewer/images/report-service/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. Right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages** and then search the package **BoldReports.Net.Core** and install to the application. The following provides detail of the packages and its usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Exports the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core**, and **Syncfusion.XlsIO.Net.Core** packages.

Syncfusion.Pdf.Net.Core | Exports the report to a PDF.

Syncfusion.DocIO.Net.Core | Exports the report to a Word.

Syncfusion.XlsIO.Net.Core | Exports the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

Newtonsoft.Json | Serializes and deserialize data for the Report Viewer. It is a mandatory package for Report Viewer, and the package version should be higher than 10.0.1 for NET Core 2.0 and others should be higher than 9.0.1.

System.Data.SqlClient | This is an optional package for Report Viewer. It should be referenced in project when the RDL report renders visual data from the SQL Server or SQL Azure data source based on RDL design. The package version should be higher than 4.1.0.

Configure Web API

The interface **IReportController** has declaration of action methods that are defined in the Web API Controller for processing the RDL, RDLC, and SSRS reports and for handling request from the Report Viewer control. The IReportController has the following action methods declaration:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportViewerController.cs**

![Adding a new controller to the project](/static/assets/javascript/report-viewer/images/report-service/add-aspnet-core-api-controller.png)

Click **Add**.

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and then implement its methods.

Create local references for the interfaces given in following table.

Interface | Purpose

IMemoryCache | Report Viewer requires a memory cache to store the information of consecutive client request and have the rendered report viewer information in server.

IHostingEnvironment | IHostingEnvironment used to get the report stream from application `wwwroot\Resources` folder.

You cannot load the application report with path information in ASP.NET Core service. So, you should load the report as stream in `OnInitReportOptions`.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-
    detail.rdl", System.IO.FileMode.Open, System.IO.FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}
'
```

You can replace the template code with the following code.

```
'csharp
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
    // Report viewer requires a memory cache to store the information of consecutive client request and
    // have the rendered report viewer information in server.

    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    // IHostingEnvironment used with sample to get the application data from wwwroot.

    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // Post action to process the report from server based json parameters and send the result back to the
    client.
```

```
public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
{
    _cache = memoryCache;
    _hostingEnvironment = hostingEnvironment;
}

// Post action to process the report from server based json parameters and send the result back to the
client.

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
{
    return ReportHelper.ProcessReport(jsonArray, this, this._cache);
}

// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-
detail.rdl", System.IO.FileMode.Open, System.IO FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}

// Method will be called when reported is loaded with internally to start to layout process with
ReportHelper.

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

//Get action for getting resources from the report
[ActionName("GetResource")]
[AcceptVerbs("GET")]

// Method will be called from Report Viewer client to get the image src for Image report item.

public object GetResource(ReportResource resource)
{
```

```
return ReportHelper.GetResource(resource, this, _cache);
}

[HttpPost]
public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, _cache);
}
}

`
```

The `sales-order-detail.rdl` report can be downloaded from [here](#). Also, you can add the report from Syncfusion installation location. For more information on installed sample location, see [Samples and demos](#).

Run the application and use the API URL in the Report Viewer `reportServiceUrl` property.

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

Enable Cross-Origin requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Call `AddCors` in `Startup.ConfigureServices` to add CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddMvc();
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
    {
        builder.AllowAnyOrigin()
            .AllowAnyMethod()
            .AllowAnyHeader();
    }));
}
```

To specify the CORS policy for Home controller, add the [EnableCors] attribute to the controller class. Specify the policy name.

```
'csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
public class ReportViewerController : Controller, IReportController
{
    public IActionResult Index()
    {
        return View();
    }
    ....
}
```

How to queries for Bold Reports ReportViewer

This section helps to get the answer for the frequently asked how to queries in Bold Reports JavaScript ReportViewer.

[Create a RDL report](#)

[Create a RDLC report](#)

[Render data visualization items](#)

[Use Report Viewer in Blazor Web Assembly \(ASP.NET Core Hosted\) application](#)

[Use Report Viewer in Blazor Server application](#)

[Change the exporting document file name based on parameter](#)

[Change the connection string datasource dynamically](#)

[Disable the vertical scrollbar in parameter panel](#)

Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

Bold Reports Web Report Designer.

Microsoft Report Builder.

Visual Studio Report Server project template.

[Bold Reports Report Designer](#)

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

[Microsoft SQL Report Builder](#)

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

[Visual Studio Report Server template](#)

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

[Create a RDLC report using business object data source](#)

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

[Prerequisites](#)

Microsoft Visual Studio 2017 or higher

[Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

[Create business object class](#)

Open Visual Studio from the File menu and select **New Project**.

Create project with class library type from the project type list.

![Add a new class library project](/static/assets/javascript/report-viewer/images/how-to/create-report/class-library-project.png)

Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
    public string ProdCat { get; set; }
```

```
public string SubCat { get; set; }  
public string OrderYear { get; set; }  
public string OrderQtr { get; set; }  
public double Sales { get; set; }  
}  
,
```

Clean and build the application.

[Add an RDLC report](#)

Right-click the project and click **Add > New Item**.

Search Report with new item and select **Report Wizard** to start the report creation with dataset selection.

Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/javascript/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

[Data source and table configuration wizard](#)

Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/javascript/report-viewer/images/how-to/create-report/choose-data-source-type.png)

Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/javascript/report-viewer/images/how-to/create-report/select-data-objects.png)

In the DataSet Properties wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/javascript/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/javascript/report-viewer/images/how-to/create-report/arrange-table-fields.png)

Choose the table layout and click **Next**.

Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/javascript/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/javascript/report-viewer/images/how-to/create-report/sales-report-design.png)

To render the RDLC using Report Viewer, refer to the [RDLC Report](#) section.

Adding data visualization scripts

To render the report with data visualization components such as chart, gauge and map items, must add scripts of the visualization element. The following table shows the script reference that need to be added in Report Viewer page for data visualization elements.

Visualization Item | Script File

Chart | ej.chart.min.js|

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item add chart control script ej.chart.min.js before the bold.report-viewer.min.js in Report Viewer page as in following code sample.

```
'html
<link
href="http://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="http://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script
src="http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="http://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

The following code can be used to render the chart, gauge and map report items in Report Viewer.

```
'html
<link
href="http://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="http://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
```

```

<script
src="http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script, only if your report contains the gauge report item. -->
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
<!--Used to render the map item. Add this script, only if your report contains the map report item.-->
<script src="http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="http://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

How to use JavaScript ReportViewer in `Blazor Web Assembly` App application

Bold Reports JavaScript ReportViewer can be used with Blazor applications using [ASP.NET Core Blazor JavaScript interoperability](#). Refer to the following steps to use Bold Reports JavaScript ReportViewer with Blazor Web Assembly App template application.

ReportViewer requires a Web API service for report processing. So, you have to create the Blazor Web Assembly App along with ASP.NET Core Hosted configuration to have the ASP.NET Core back end with Blazor application itself.

Refer a scripts and CSS

Add the following styles and scripts in client project wwwroot\index.html.

```

`html
<link href="https://cdn.syncfusion.com/2.2.32/bold-
reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://cdn.syncfusion.com/js/assets/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
```

```

<script src="https://cdn.syncfusion.com/2.2.32/bold-
reports/common/bold.reports.common.min.js"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-
reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-
visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-visualization/ej2-
circulargauge.min.js"></script>
<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/bold.report-viewer.min.js"></script>
`
```

Adding a ReportViewer

Add the following method in `index.html` to add our [JavaScript ReportViewer](#) with element using jQuery. This will be invoked using [IJSRuntime](#) from razor pages along with name of the report and element need to be created with ReportViewer.

```

`html
<script>

function RenderReportViewer (reportPathInfo, elementID) {
    $("#" + elementID).boldReportViewer({
        reportServiceUrl: '/api/ReportApi',
        reportPath: reportPathInfo
    });
}

</script>
```

`

Create div tag and code with razor page where you want to render the ReportViewer.

```
'csharp
@page "/reporting"
@using Microsoft.JSInterop
@using Microsoft.AspNetCore.Components
@inject IJSRuntime JSRuntime

<div id="reportViewerControl" style="width: 100%;height:800px" />

@code {
    private string reportPathInput = "Report1";
    // You have to call this method based on your need. It can called while loading the page or after
    selection the report.

    public async void RenderReport()
    {
        await JSRuntime.InvokeVoidAsync("RenderReportViewer", reportPathInput , "reportViewerControl");
    }
    /// https://docs.microsoft.com/en-us/aspnet/core/blazor/components?view=aspnetcore-3.0
    ///
    https://github.com/aspnet/AspNetCore.Docs/tree/master/aspnetcore/blazor/common/samples/3.x/BlazorSample

    protected override void OnAfterRender(bool firstRender)
    {
        // If you want to load in intialization you can make use of this.
        //RenderReport();
    }
}
```

Create a Web API service

The Report Viewer requires a Web API service to process the report files, and you should create ASP.NET Core Web API Service for server interaction and do the processing in API using Report Helper.

Refer [ASP.NET Core Web API Service](#)

You can ignore Enable Cross origin topic with documentation. since, the service with Blazor application and there is no need to enable cores.

Also, ensure the following steps while creating the Web API service.

You should route the API properly for interaction with attribute
[Route("api/{controller}/{action}/{id?}").

[FromBody] should be used for PostReportAction for action method.

[HttpPost] and [HttpGet] attributes should be used in respective ReportViewer interaction methods.

You will get compatibility issues with ReportViewer service, which is created from default template.
So, following changes required in application StartUp is based on application template type.

You will get the [400 Bad request](#) with service interaction due to the validation of null values with new version of ASP.NET Core. To avoid this issue, you have to remove the [ApiController] attribute from Report API service. Otherwise, you have to set SuppressModelStateInvalidFilter to true in ConfigureApiBehaviorOptions to avoid this bad request. API behavior options need to updated in ConfigureServices method in Startup.cs of server project.

`csharp

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddMvc().AddNewtonsoftJson()
    .ConfigureApiBehaviorOptions(options =>
    {
        options.SuppressModelStateInvalidFilter = true;
    });
    services.AddResponseCompression(opts =>
    {
        opts.MimeTypes = ResponseCompressionDefaults.MimeTypes.Concat(
            new[] { "application/octet-stream" });
    });
}
```

The following option will be required to avoid the bad request. If you are making the file download application with your Blazor application using `location.rel='nofollow' href`.

```
'csharp
options.SuppressConsumesConstraintForFormFileParameters = true;
options.SuppressInferBindingSourcesForParameters = true;
'
```

How to use Bold Reports JavaScript ReportViewer with `Blazor Server` App

Bold Reports JavaScript ReportViewer can be used with Blazor applications using [ASP.NET Core Blazor JavaScript interoperability](#). Refer to the following steps to use Bold Reports JavaScript ReportViewer with Blazor Server App template application.

Refer a scripts and CSS

Add the following styles and scripts `_host.cshtml` available in pages folder.

```
'html
<link href="https://cdn.syncfusion.com/2.2.32/bold-
reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://cdn.syncfusion.com/js/assets/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-
reports/common/bold.reports.common.min.js"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-
reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-
visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-visualization/ej2-
circulargauge.min.js"></script>
```

```
<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="https://cdn.syncfusion.com/2.2.32/bold-reports/bold.report-viewer.min.js"></script>
`
```

Adding a ReportViewer

Add the following method in `_host.cshtml` to add our [JavaScript ReportViewer](#) with element using jQuery. This will be invoked using [IJSRuntime](#) from razor pages along with name of the report and element need to be created with ReportViewer.

```
`html
<script>
function RenderReportViewer (reportPathInfo, elementID) {
    $("#" + elementID).boldReportViewer({
        reportServiceUrl: '/api/ReportApi',
        reportPath: reportPathInfo
    });
}
</script>
`
```

Create div tag and code with razor page where you want to render the Report Viewer.

```
`csharp
@page "/reporting"
@using Microsoft.JSInterop
@using Microsoft.AspNetCore.Components
@inject IJSRuntime JSRuntime
<div id="reportViewerControl" style="width: 100%;height:800px" />
@code {
    private string reportPathInput = "Report1";
    // You have to call this method based on your need. It can be called while loading the page or after
    // selection the report.
    public async void RenderReport()
    {
```

```

await JSRuntime.InvokeVoidAsync("RenderReportViewer", reportPathInput, "reportViewerControl");
}

/// https://docs.microsoft.com/en-us/aspnet/core/blazor/components?view=aspnetcore-3.0
///
https://github.com/aspnet/AspNetCore.Docs/tree/master/aspnetcore/blazor/common/samples/3.x/BlazorSample

protected override void OnAfterRender(bool firstRender)
{
    // If you want to load in initialization you can make use of this.
    //RenderReport();
}
}
`
```

Create a Web API service

The ReportViewer requires a Web API service to process the report files, and you should create ASP.NET Core Web API Service for server interaction and do the processing in API using Report Helper.

Refer [ASP.NET Core Web API Service](#)

You can ignore Enable Cross origin topic with documentation. since, the service with Blazor application and there is no need to enable cores.

Also, ensure the following steps while creating the Web API service.

You should route the API properly for interaction with attribute
`[Route("api/{controller}/{action}/{id?}")]`.

`[FromBody]` should be used for `PostReportAction` for action method.

`[HttpPost]` and `[HttpGet]` attributes should be used in respective report viewer interaction methods.

You will get compatibility issues with ReportViewer service, which is created from default template. So, following changes required in application StartUp is based on application template type.

By default, this template will not map controller in your application. So, you have to map controller manually using `MapControllers()` in app end points as follows in `startup.cs`.

```

`csharp
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
```

```
{  
if (env.IsDevelopment())  
{  
    app.UseDeveloperExceptionPage();  
}  
else  
{  
    app.UseExceptionHandler("/Error");  
    // The default HSTS value is 30 days. You may want to change this for production scenarios, see  
    // https://aka.ms/aspnetcore-hsts.  
    app.UseHsts();  
}  
app.UseHttpsRedirection();  
app.UseStaticFiles();  
app.UseRouting();  
app.UseEndpoints(endpoints =>  
{  
    endpoints.MapControllers();  
    endpoints.MapBlazorHub();  
    endpoints.MapFallbackToPage("/_Host");  
});  
}  
`
```

By default, Json serialization will not be used in the service interaction. For this template, add the `AddNewtonsoftJson()` with service as follows. By default, you will not get `AddNewtonsoftJson()` extension method with service, you need to add a package reference `Microsoft.AspNetCore.Mvc.NewtonsoftJson` to get this extension method.

```
'csharp  
public void ConfigureServices(IServiceCollection services)  
{  
    services.AddRazorPages();  
    services.AddServerSideBlazor();  
    services.AddSingleton<WeatherForecastService>();  
    services.AddMvc().AddNewtonsoftJson();
```

```

    }
    '

```

How to change the exporting document file name based on parameter

Find the following steps to change the file based on parameter values in Report.

Create the file exporting file name using the parameters in **onRenderingComplete** event and store it in local variable.

```

`html
function onRenderingComplete(event) {
    var parameters = event.reportParameters;
    if(parameters){
        for (var i = 0; i < parameters.length; i++) {
            if(parameters[i].Name == "Department"){
                this.exportFileName = "Sales for " + parameters[i].Value;
            }
        }
    }
    '

```

Use the file Name property with export, click event to change the file using the value stored in local variable used for having the file using the parameters.

```

`html
function onExportItemClick(event) {
    event.fileName = this.exportFileName ;
}
'

```

How to change the connection string datasource dynamically

Find the following steps to change connection string in datasource.

You need a report action information to get the data source information from report. So, stored the JsonResult information to local property in **PostReportAction** method as shown in the following code example.

```

`csharp
private Dictionary<string, object> _jsonResult;
//Post action for processing the rdl/rdlc report

```

```
public object PostReportAction(Dictionary<string, object> jsonResult)
{
if (jsonResult != null && jsonResult.Keys.Count > 0)
{
    _jsonResult = jsonResult;
}
return ReportHelper.ProcessReport(jsonResult, this);
}
```

Use the JsonResult information with **ReportHelper.GetDatasource** API to get the data source details of the reports and you have to use the **DataSourceCredentials** to change the connection string of the data source.

```
`csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (jsonResult != null && jsonResult.Keys.Count > 0)
{
List<DataSourceInfo> datasources = ReportHelper.GetDataSources(_jsonResult, this);
foreach (DataSourceInfo item in datasources)
{
if (item.DataProvider == "SQL")
{
string connectionString = "Data Source = dataplatformdemodata.syncfusion.com; Initial Catalog =
AdventureWorks; User ID = 'demoreadonly@data-platform-demo'; Password = 'N@c)=Y8s*1&dh'";
DataSourceCredentials DataSourceCredentials = new DataSourceCredentials();
DataSourceCredentials.Name = item.DataSourceName;
DataSourceCredentials.UserId = null;
DataSourceCredentials.Password = null;
DataSourceCredentials.ConnectionString = connectionString;
DataSourceCredentials.IntegratedSecurity = false;//if windows credentials means we need to pass true
as IntegratedSecurity
reportOption.ReportModel.DataSourceCredentials = new List<DataSourceCredentials>
{
DataSourceCredentials
```

```

};

}

}

}

}

`
```

How to disable the vertical scrollbar in parameter panel

To disable the vertical scrollbar in parameter panel, set the `enableparameterblockscroller` property to false.

```

<span style="font-weight:bold">Example</span>

`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
enableParameterBlockScroller: false
});
</script>
`
```

JavaScript Report Viewer API reference

Using the Report Viewer public properties, events, and members you can change the report options programmatically. The API gives you simple access to the functionality behind the Report Viewer. You can use this to create your own custom applications or to script interactions with Report Viewer. In this documentation contains the Report Viewer public API's details with code snippets.

[Properties](#)

[Events](#)

[Members](#)

[Methods](#)

Members

```

<h3 class="doc-prop-wrapper" id="datasources" data-Path="datasources-dataSources">
<a href="#datasources" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41- .91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h- 1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>dataSources</span>
<span class="doc-prop-type"> dataSources
</span>
</h3>
Gets or sets the list of data sources for the RDLC report.
Defaults to []
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$( "#reportviewer" ).boldReportViewer({ dataSources: [{name:"Menu Items",
values:[{ OrderId: "21D60", FoodName: "Burger", Price: 20, Category: "Veg" },
{ OrderId: "21D61", FoodName: "Pizza", Price: 25, Category: "Non-Veg" },
{ OrderId: "21D63", FoodName: "Sandwiches", Price: 30, Category: "Non-Veg" },
{ OrderId: "21D65", FoodName: "Chicken Drum Sticks", Price: 23, Category: "Non-Veg" },
{ OrderId: "21D64", FoodName: "Fulka", Price: 15, Category: "Veg" }]}]
});
</script>
` 

<h3 class="doc-prop-wrapper" id="enablepagecache" data-Path="enablepagecache-enablePageCache">
<a href="#enablepagecache" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41- .91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h- 1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>enablePageCache</span>
<span class="doc-prop-type"> boolean
</span>
```

```
</h3>

Enables or disables the page cache of report.

Defaults to false

<span style="font-weight:bold">Example</span>

`js

<div id="reportviewer"></div>

<script>

$("#reportviewer").boldReportViewer({ enablePageCache: false });

</script>
`


<h3 class="doc-prop-wrapper" id="exportsettings" data-Path="exportsettings-exportSettings">

<a href="#exportsettings" aria-hidden="true" class="anchor">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>

</svg>

</a><span class='doc-prop-name'>exportSettings</span>

<span class="doc-prop-type"> exportSettings
</span>

</h3>

Specifies the export settings.

<span style="font-weight:bold">Example</span>

`js

<div id="reportviewer"></div>

<script>

$("#reportviewer").boldReportViewer(
{
    exportSettings:{ excelFormat: ej.ReportViewer.ExcelFormats.Excel97to2003,
        wordFormat: ej.ReportViewer.WordFormats.Doc
    }
});

</script>
```

```
`  
  
<h3 class="doc-prop-wrapper" id="isresponsive" data-Path="isresponsive-isResponsive">  
  <a href="#isresponsive" aria-hidden="true" class="anchor">  
    <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
      <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
    </svg>  
  </a><span class='doc-prop-name'>isResponsive</span>  
  <span class="doc-prop-type"> boolean  
  </span>  
</h3>
```

When set to true, adapts the report layout to fit the screen size of devices on which it renders.

Defaults to *true*

```
<span style="font-weight:bold">Example</span>  
'js  
<div id="reportviewer"></div>  
<script>  
  $('#reportviewer').boldReportViewer({ isResponsive: true });  
</script>  
'
```

```
<h3 class="doc-prop-wrapper" id="locale" data-Path="locale-locale">  
  <a href="#locale" aria-hidden="true" class="anchor">  
    <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
      <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
    </svg>  
  </a><span class='doc-prop-name'>locale</span>  
  <span class="doc-prop-type"> number  
  </span>  
</h3>
```

Specifies the locale for report viewer.

Defaults to *en-US*

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  locale: "it-IT"
});
</script>
` 

<h3 class="doc-prop-wrapper" id="pagesettings" data-Path="pagesettings-pageSettings">
<a href="#pagesettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>pageSettings</span>
<span class="doc-prop-type"> pageSettings
</span>
</h3>


Specifies the page settings.


<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  pageSettings:{ paperSize: ej.ReportViewer.PaperSize.A4,
    height: 11.69,
    width: 8.27
  }
}
```

```
});

</script>
` 

<h3 class="doc-prop-wrapper" id="parameters" data-Path="parameters-parameters">
<a href="#parameters" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>parameters</span>
<span class="doc-prop-type"> parameters
</span>
</h3>

Gets or sets the list of parameters associated with the report.

Defaults to []
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({
parameters: [
name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
}]
});
</script>
` 

<h3 class="doc-prop-wrapper" id="toolbarsettings" data-Path="toolbarsettings-toolbarSettings">
<a href="#toolbarsettings" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>toolbarSettings</span>
<span class="doc-prop-type"> toolbarSettings
</span>
</h3>


Specifies the toolbar settings.


<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  toolbarSettings:{ showTooltip: true }
});
</script>
`


<h3 class="doc-prop-wrapper" id="parametersettings" data-Path="parametersettings-parameterSettings">
<a href="#parametersettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>parameterSettings</span>
<span class="doc-prop-type"> parameterSettings
</span>
</h3>


Specifies the parameter settings.


```

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$( "#reportviewer" ).boldReportViewer(
{
parameterSettings: {
delimiterChar: ",",
popupHeight: "200px",
popupWidth: "150px",
itemWidth: '250px',
labelWidth: 'auto',
attributes: { 'autocomplete': 'off' },
hideParameterBlock: false
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="printmode" data-Path="printmode-printMode">
<a href="#printmode" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>printMode</span>
<span class="doc-prop-type"> boolean
</span>
</h3>


Enables and disables the print mode.



Defaults to false


<span style="font-weight:bold">Example</span>
`js
```

```

<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
printMode:true
});
</script>
`<h3 class="doc-prop-wrapper" id="printoption" data-Path="printoption-printOption">
<a href="#printoption" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>printOption</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.PrintOptions" style = "display:none"></ts>

```

Specifies the print option of the report.

| Name | Description |
|---------|-------------------------------------------------|
| Default | Specifies the Default property in printOptions. |
| NewTab | Specifies the NewTab property in printOptions. |
| None | Specifies the None property in printOptions. |

Defaults to *ej.ReportViewer.PrintOptions.Default*

```

<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({ printOption: ej.ReportViewer.PrintOptions.Default });
</script>

```

```

`<h3 class="doc-prop-wrapper" id="processingmode" data-Path="processingmode-processingMode">
<a href="#processingmode" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>processingMode</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.ProcessingMode" style = "display:none"></ts>
```

Specifies the processing mode of the report.

| Name | Description |
|--------|--------------------------------------------------|
| Remote | Specifies the Remote property in processingMode. |
| Local | Specifies the Local property in processingMode. |

Defaults to *ej.ReportViewer.ProcessingMode.Remote*

```

<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({ processingMode: ej.ReportViewer.ProcessingMode.Remote });
</script>
`<h3 class="doc-prop-wrapper" id="rendermode" data-Path="rendermode-renderMode">
<a href="#rendermode" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>renderMode</span>
<span class="doc-prop-type"> enum
</span>
</h3>
```

<ts name = "ej.ReportViewer.RenderMode" style = "display:none"></ts>

Specifies the render layout.

| Name | Description |
|---------|-----------------------------------------------------------------------|
| Default | Specifies the Default property in RenderMode to get default output. |
| Mobile | Specifies the Mobile property in RenderMode to get specified output. |
| Desktop | Specifies the Desktop property in RenderMode to get specified output. |

Defaults to *ej.ReportViewer.RenderMode.Default*

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({ renderMode: ej.ReportViewer.RenderMode.Default });
</script>
`
```

```
<h3 class="doc-prop-wrapper" id="reportpath" data-Path="reportpath-reportPath">
<a href="#reportpath" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>reportPath</span>
```

 string

</h3>

Gets or sets the path of the report file.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({ reportPath: "~/App_Data/Sample.rdl" });
</script>
` 

<h3 class="doc-prop-wrapper" id="reportserverurl" data-Path="reportserverurl-reportServerUrl">
<a href="#reportserverurl" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>reportServerUrl</span>
<span class="doc-prop-type"> string
</span>
</h3>
Gets or sets the reports server URL.
Defaults to empty
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({ reportServerUrl: "http://mvc.syncfusion.com/reportserver" });
</script>
` 

<h3 class="doc-prop-wrapper" id="reportserviceurl" data-Path="reportserviceurl-reportServiceUrl">
<a href="#reportserviceurl" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>reportServiceUrl</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

```
</svg>
</a><span class='doc-prop-name'>reportServiceUrl</span>
string
</span>
</h3>


Specifies the report Web API service URL.



Defaults to empty



<span style="font-weight:bold">Example</span>


`js
<div id="reportviewer"></div>
<script>
$( "#reportviewer" ).boldReportViewer({ reportServiceUrl: "../api/RDLReport" });
</script>
`
```

<h3 class="doc-prop-wrapper" id="zoomfactor" data-Path="zoomfactor-zoomFactor">

zoomFactor

number

</h3>

Gets or sets the zoom factor for report viewer.

Defaults to 1

Example

`js

```
<div id="reportviewer"></div>
<script>
$( "#reportviewer" ).boldReportViewer({ zoomFactor: 2 });
</script>
```

'

```
<h3 class="doc-prop-wrapper" id="serviceauthorizationtoken" data-Path="serviceauthorizationtoken-serviceAuthorizationToken">
<a href="#serviceauthorizationtoken" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>serviceAuthorizationToken</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Specifies the token for authorizing reporting service URL to process the reports.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$(function () {
var dataValue = "";
var apiRequest = new Object();
apiRequest.password = "demo";
apiRequest.userid = "guest";
$.ajax({
type: "POST",
url: "http://reportserver.syncfusion.com/api/get-user-key",
data: apiRequest,
success: function (data) {
dataValue = data.Token;
var token = JSON.parse(dataValue);
$("#report viewer").boldReportViewer(
{

```

```
reportServiceUrl: "http://reportserver.syncfusion.com/ReportService/api/Viewer",
serviceAuthorizationToken: token["tokentype"] + " " + token["accesstoken"],
reportPath: '/Sample Reports/Company Sales'
});
}
});
});
});
</script>
`  
  

<h3 class="doc-prop-wrapper" id="enableparameterblocksroller" data-  
Path="enableparameterblocksroller-enableParameterBlockScroller">  

<a href="#enableparameterblocksroller" aria-hidden="true" class="anchor">  

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-  
1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  

</svg>  

</a><span class='doc-prop-name'>enableParameterBlockScroller</span>  

<span class="doc-prop-type"> boolean  

</span>  

</h3>  


Enables and disables the parameter block scroller.



Defaults to true



<span style="font-weight:bold">Example</span>


`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
enableParameterBlockScroller: false
});
</script>
`
```

```
<h3 class="doc-prop-wrapper" id="enabledatasourceblockscroller" data-Path="enabledatasourceblockscroller-enableDatasourceBlockScroller">
<a href="#enabledatasourceblockscroller" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>enableDatasourceBlockScroller</span>
<span class="doc-prop-type"> boolean
</span>
</h3>


Enables and disables the data source credential block scroller.



Defaults to true


<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$( "#report viewer" ).boldReportViewer(
{
  enableDatasourceBlockScroller: false
});
</script>
`


<h3 class="doc-prop-wrapper" id="sizetoreportcontent" data-Path="sizetoreportcontent-sizeToReportContent">
<a href="#sizetoreportcontent" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>sizeToReportContent</span>
<span class="doc-prop-type"> boolean
```

```
</span>
</h3>

Render the Report Viewer height based on the report content size.

Defaults to false

Example
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
sizeToReportContent: true
});
</script>
`


<h3 class="doc-prop-wrapper" id="autorender" data-Path="autorender-autoRender">
<a href="#autorender" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>autoRender</span>
<span class="doc-prop-type"> boolean
</span>
</h3>

Enables and disables the rendering of Viewer when default values are specified for the parameters.

Defaults to true

Example
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
```

```
autoRender: false
});

</script>
` 

<h3 class="doc-prop-wrapper" id="enablenotificationbar" data-Path="enablenotificationbar-enableNotificationBar">

<a href="#enablenotificationbar" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>enableNotificationBar</span>
<span class="doc-prop-type"> boolean
</span>
</h3>

Enables and disables the Error Notification bar.

Defaults to true

<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
  enableNotificationBar: false
});
</script>
` 

<h3 class="doc-prop-wrapper" id="enabledropdownsearch" data-Path="enabledropdownsearch-enableDropDownSearch">

<a href="#enabledropdownsearch" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>  
</a><span class='doc-prop-name'>enableDropDownSearch</span>  
<span class="doc-prop-type"> boolean  
</span>  
</h3>
```

Enables and disables the drop-down parameter search.

Defaults to *false*

```
<span style="font-weight:bold">Example</span>
```

```
`js
```

```
<div id="report viewer"></div>
```

```
<script>
```

```
$("#report viewer").boldReportViewer(
```

```
{
```

```
enableDropDownSearch: true
```

```
</script>
```

```
,
```

```
<h3 class="doc-prop-wrapper" id="enablepagevirtualization" data-Path="enablepagevirtualization-  
enablePageVirtualization">
```

```
<a href="#enablepagevirtualization" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-  
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>enablePageVirtualization</span>
```

```
<span class="doc-prop-type"> boolean  
</span>
```

```
</h3>
```

Enables and disables the page virtualization.

Defaults to *false*

```
<span style="font-weight:bold">Example</span>
```

```
'js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
enablePageVirtualization: true
});
</script>
` 

<h3 class="doc-prop-wrapper" id="waitingpopuptemplate" data-Path="waitingpopuptemplate-waitingpopuptemplate">
<a href="#waitingpopuptemplate" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>waitingPopupTemplate </span>
<span class="doc-prop-type"> String
</span>
</h3>
Gets or sets the waiting popup template for the Report viewer.
Defaults to null
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
waitingPopupTemplate: '<div id="customLoadingtemplate" style="background-image:url(&lt;https://js.syncfusion.com/demos/web/content/images/waitingpopup/load_light.gif&gt;); width: 200px; height: 15px"><p>Loading report... </p></div>'
});
</script>
```

Methods destroy()

Methods

[destroy\(\)](#)

Destroy the client and server side report viewer processing objects.

Example

'js

```
<div id="report viewer"></div>
```

```
<script>
```

```
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
```

```
reportviewerObj.destroy(); //Destroy the report viewer processing objects.
```

```
</script>
```

'

[exportReport\(\)](#)

Export the report to the specified format.

Example

'js

```
<div id="reportviewer">ReportViewer</div>
```

```
<script>
```

```
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
```

```
reportviewerObj.exportReport("PDF"); //Exports the report into PDF format.
```

```
</script>
```

'

[fitToPage\(\)](#)

Fit the report page to the container.

Example

'js

```
<div id="reportviewer">ReportViewer</div>
```

```
<script>
```

```
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
```

```
reportviewerObj.fitToPage(); // To fit the report page.
```

```
</script>
```

'

| | |
|---------|-------------------|
| Methods | fitToPageHeight() |
|---------|-------------------|

fitToPageHeight()

Fit the report page height to the container.

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.fitToPageHeight(); // To fit the report page height.
</script>
`
```

fitToPageWidth()

Fit the report page width to the container.

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.fitToPageWidth(); // To fit the report page width.
</script>
`
```

getDataSetNames()

Get the available datasets name of the rdlc report.

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.getDataSetNames(); // To get the dataset names.
</script>
`
```

getParameters()

Get the available parameters of the report.

```
<span style="font-weight:bold">Example</span>
`js
```

Methods [gotoFirstPage\(\)](#)

```
<div id="reportviewer">ReportViewer</div>  
  
<script>  
var reportviewerObj = $("#reportviewer").data("boldReportViewer");  
  
reportviewerObj.getParameters(); // To get the parameters.  
</script>
```

gotoFirstPage()

Navigate to first page of report.

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.gotoFirstPage(); // To navigate to first page
</script>
```

gotoLastPage()

[Navigate to last page of the report.](#)

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.gotoLastPage(); // Navigate to the last page
</script>
```

gotoNextPage()

Next Page

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.gotoNextPage(); //To navigate to the next page
```


Methods

refresh()

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.printLayout(); //Changes between print layout and normal modes.
</script>
`
```

refresh()

Refresh the report.

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.refresh(); // To refresh the report.
</script>
`
```

setParameterBlockVisibility()

Specify the visibility of parameter block.

| Parameter | Type | Description |
|-----------|---------|---------------------------------------------------------------------------------------------------------------------------------|
| show | boolean | Indicates whether the parameter block needs to be shown or not. true if you have to show the parameter block; otherwise, false. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.setParameterBlockVisibility(true);
</script>
`
```

cancelRendering()

Cancel the report processing.

Events

drillThrough

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer">ReportViewer</div>
<script>
var reportviewerObj = $("#reportviewer").data("boldReportViewer");
reportviewerObj.cancelRendering(); // To cancel the report processing.
</script>
`
```

Events

drillThrough

Fires during drill through action done in report. If you want to perform any operation when a drill through action is performed, you can make use of the drillThrough event.

| Name | Type | Description |
|------------|---------|---------------------------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| actionInfo | Object | returns the actionInfo's parameters bookmarkLink, reportName, parameters. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
drillThrough: function (args) {
// Write a code block to perform any operation when drill through action occurs in report.
}
});
</script>
`
```

renderingBegin

Fires before report rendering is completed. If you want to perform any operation before the rendering of report, you can make use of the renderingBegin event.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
//rendering begin event for report.
$("#report viewer").boldReportViewer({
renderingBegin:function(args) {
// Write a code block to perform any operation before rendering.
}
});
</script>
`
```

renderingComplete

Fires after report rendering completed. If you want to perform any operation after the rendering of report,you can make use of this renderingComplete event.

| Name | Type | Description |
|------------------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |
| reportParameters | object | returns the collection of parameters. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
//rendering complete event for report viewer control.
$("#report viewer").boldReportViewer({
renderingComplete:function(args) {
```

```
// Write a code block to perform any operation after rendering completed.
}
});
</script>
`
```

reportError

Fires when any error occurred while rendering the report. If you want to perform any operation when an error occurs in the report, you can make use of the reportError event.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| error | string | returns the error details. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
reportError: function (args) {
// Write a code block to perform any operation when report error occurs.
}
});
</script>
`
```

reportExport

Fires when the report is being exported. If you want to perform any operation before exporting of report, you can make use of the reportExport event.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

Events

reportLoaded

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
reportExport: function (args) {
// Write a code block to perform any action before exporting of report.
}
});
</script>
`
```

reportLoaded

Fires when the report is loaded. If you want to perform any operation after the successful loading of report, you can make use of the reportLoaded event.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
reportLoaded: function(args) {
// Write a code block to perform any action when the report is loaded successfully.
}
});
</script>
`
```

showError

Fires when user clicks on a failed report item in the rendered report, before displaying error details dialog. If you want to show custom error detail or perform any action before viewing error detail, you can make use of the showError event.

| Name | Type | Description |
|-----------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| errorCode | string | returns the error code. |
| message | string | returns the error message. |
| detail | string | returns the detailed error information. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
showError: function (args) {
// Write a code block to perform any operation when user clicks a failed item in a report.
}
});
</script>
`
```

[viewReportClick](#)

Fires when click the View Report Button.

| Name | Type | Description |
|------------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| parameters | Object | returns the parameter collection. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
viewReportClick: function (args) {
// Write a code block to perform any operation after destroy of report viewer.
}
});
```

Events ajaxBeforeLoad

```
}
```

```
});
```

```
</script>
```

```
,
```

ajaxBeforeLoad

Fires before the AJAX request process started.

| Name | Type | Description |
|---------------------------|--------|----------------------------------------|
| reportViewerToken | string | returns the reportViewerToken. |
| serviceAuthorizationToken | string | returns the serviceAuthorizationToken. |
| headerReq | Object | Send the header request collection. |
| headers | Object | Send the headers collection. |
| data | string | Send the custom data. |

Example

```
`js
```

```
<div id="report viewer"></div>
```

```
<script>
```

```
$("#report viewer").boldReportViewer({
```

```
ajaxBeforeLoad: function (args) {
```

```
// Write a code block to perform any operation after destroy of report viewer.
```

```
}
```

```
});
```

```
</script>
```

```
,
```

ajaxSuccess

Fires when AJAX post call succeed.

| Name | Type | Description |
|------|--------|---------------------------|
| data | object | returns the success data. |

Example

```
`js
```

```
<div id="report viewer"></div>
```

```
<script>
```

Events

ajaxError

```
$("#report viewer").boldReportViewer({  
    ajaxSuccess: function (args) {  
        // Write a code block to perform any operation after destroy of report viewer.  
    }  
});  
</script>  
`
```

ajaxError

Fires when AJAX request failed.

| Name | Type | Description |
|------|--------|---------------------------|
| msg | string | returns the error details |

```
<span style="font-weight:bold">Example</span>  
`js  
<div id="report viewer"></div>  
<script>  
$("#report viewer").boldReportViewer({  
    ajaxError: function (args) {  
        // Write a code block to perform any operation after destroy of report viewer.  
    }  
});  
</script>  
`
```

toolbarRendering

This event will be triggered on rendering the Report Viewer toolbar.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |
| target | JQuery | returns the toolbar container. |

```
<span style="font-weight:bold">Example</span>  
`js
```

```
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
toolbarRendering: function(args) {
// Write a code block to customize the report viewer toolbar.
}
});
</script>
`
```

[exportProgressChanged](#)

Fires when the export progress is changed. To perform any operation when the export progress is changed, use the `exportProgressChanged` event.

| Name | Type | Description |
|----------------------|---------|-------------------------------------------------------------|
| format | string | returns the export format |
| stage | string | returns the stage of export processing. |
| handled | boolean | true if the event should be handled; otherwise, false. |
| containerId | string | returns Report Viewer container Id. |
| waitingPopupTemplate | string | Set the <code>waitingPopupTemplate</code> in report viewer. |

[Example](#)

```
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
exportProgressChanged : function(args) {
// Write a code block to perform any action when the export progress changed.
}
});
</script>
`
```

[printProgressChanged](#)

Fires when the print progress is changed. To perform any operation when the print progress is changed, use the `printProgressChanged` event.

| Name | Type | Description |
|----------------------|---------|--------------------------------------------------------|
| stage | string | returns the stage of export processing. |
| currentPage | string | returns the currentPage value |
| totalPages | string | returns the totalPages value |
| handled | boolean | true if the event should be handled; otherwise, false. |
| containerId | string | returns Report Viewer container Id. |
| waitingPopupTemplate | string | Set the waitingPopupTemplate in report viewer. |

Example

```
'js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
printProgressChanged : function(args) {
// Write a code block to perform any action when the print progress changed.
}
});
</script>
`
```

exportItemClick

Fires when the export items are clicked. To perform any operation when the export items are clicked, use the exportItemClick event.

| Name | Type | Description |
|-------|--------|----------------------------------|
| value | string | returns the export format value. |

Example

```
'js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
exportItemClick : function(args) {
// Write a code block to perform any action when the export item clicked.
}
});
```

Events toolBarItemClick

```
}
```

```
});
```

```
</script>
```

```
,
```

toolBarItemClick

Fires when the toolbar items are clicked. To perform any operation when the toolbar items are clicked, use the toolBarItemClick event.

| Name | Type | Description |
|------------|--------|-----------------------------------------------------------|
| target | string | returns the toolbar clicked item name . |
| cssClass | string | returns the CSS class name specified for the toolbar item |
| groupIndex | string | returns the Toolbar item rendered group index |
| Index | string | returns the Toolbar item rendered index |
| value | string | returns the Toolbar item value. |

```
<span style="font-weight:bold">Example</span>
```

```
`js
```

```
<div id="report viewer"></div>
```

```
<script>
```

```
$("#report viewer").boldReportViewer({
```

```
toolBarItemClick : function(args) {
```

```
// Write a code block to perform any action when the toolbar item clicked.
```

```
}
```

```
});
```

```
</script>
```

```
,
```

hyperlink

Fires when the hyperlink action is performed in the report. To perform any operation during the hyperlink action, use the hyperlink event.

| Name | Type | Description |
|------------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| actionInfo | Object | returns the actionInfo's hyperLink detail |
| model | object | returns the report model. |

| Name | Type | Description |
|------|--------|--------------------------------|
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
hyperlink: function (args) {
// Write a code block to perform any operation when hyperlink action occurs in report.
}
});
</script>
`
```

reportPrint

Fires when the report print action is performed in the report. To perform any operation during the report print action, use the ReportPrint event.

| Name | Type | Description |
|-------------|---------|---------------------------------------------------------------------|
| isStyleLoad | boolean | true if you have to load the external style file; otherwise, false. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer({
reportPrint : function(args) {
// Write a code block to perform any action when the export item clicked.
}
});
</script>
`
```

JavaScript Report Viewer Properties

This section provides the JavaScript Report Viewer properties with example code snippets used to customize the report preview, export, and much more.

[Page Settings](#)

[Parameters](#)

[Export Settings](#)

[Toolbar Settings](#)

[Parameter Settings](#)

[Data Sources](#)

pageSettings

Properties

```
<h3 class="doc-prop-wrapper" id="orientation" data-Path="orientation-orientation">
<a href="#orientation" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>orientation</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.Orientation" style = "display:none"></ts>
```

Specifies the print layout orientation.

| Name | Description |
|-----------|---------------------------------------------------------------------------------------|
| Landscape | Specifies the Landscape property in pageSettings.orientation to get specified layout. |
| portrait | Specifies the portrait property in pageSettings.orientation to get specified layout. |

Defaults to *null*

Example

```

`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
pageSettings:{ orientation: ej.ReportViewer.Orientation.Landscape }
});
</script>
` 

<h3 class="doc-prop-wrapper" id="pagesize" data-Path="pagesize-paperSize">
<a href="#pagesize" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>paperSize</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.PaperSize" style = "display:none"></ts>

```

Specifies the paper size of print layout.

| Name | Description |
|------------------|--------------------------------------------------------------------------------------|
| A3 | Specifies the A3 as value in pageSettings.paperSize to get specified size. |
| portrait | Specifies the A4 as value in pageSettings.paperSize to get specified size. |
| B4_JIS | Specifies the B4(JIS) as value in pageSettings.paperSize to get specified size. |
| B5_JIS | Specifies the B5(JIS) as value in pageSettings.paperSize to get specified size. |
| Envelope_10 | Specifies the Envelope #10 as value in pageSettings.paperSize to get specified size. |
| Envelope_Monarch | Specifies the Envelope as value in pageSettings.paperSize to get specified size. |
| Executive | Specifies the Executive as value in pageSettings.paperSize to get specified size. |
| Legal | Specifies the Legal as value in pageSettings.paperSize to get specified size. |
| Letter | Specifies the Letter as value in pageSettings.paperSize to get specified size. |

| Name | Description |
|---------|---------------------------------------------------------------------------------|
| Tabloid | Specifies the Tabloid as value in pageSettings.paperSize to get specified size. |
| Custom | Specifies the Custom as value in pageSettings.paperSize to get specified size. |

Defaults to *null*

Example

'js

<div id="reportviewer"></div>

<script>

\$("#reportviewer").boldReportViewer(

{

pageSettings:{ paperSize: ej.ReportViewer.PaperSize.A4 }

});

</script>

'

<h3 class="doc-prop-wrapper" id="height" data-Path="height-height">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>

</svg>

height

 number

</h3>

Specifies the height of print layout.

Defaults to 0

Example

'js

<div id="reportviewer"></div>

<script>

\$("#reportviewer").boldReportViewer(

```
{  
pageSettings: { height: 11.69 }  
});  
</script>  
  
<h3 class="doc-prop-wrapper" id="width" data-Path="width-width">  
<a href="#width" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>width</span>  
<span class="doc-prop-type"> number  
</span>  
</h3>  
Specifies the width of print layout.  
Defaults to 0  
<span style="font-weight:bold">Example</span>  
'js  
<div id="reportviewer"></div>  
<script>  
$("#reportviewer").boldReportViewer(  
{  
pageSettings: { width: 8.27 }  
});  
</script>  
  
<h3 class="doc-prop-wrapper" id="margins" data-Path="margins-margins">  
<a href="#margins" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

parameters `array`

Properties

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>margins</span>  
<span class="doc-prop-type"> number  
</span>  
</h3>
```

Specifies the margins of print layout.

| Name | Description |
|--------|-----------------------------------------------|
| top | Specifies the top margins of print layout. |
| right | Specifies the right margins of print layout. |
| bottom | Specifies the bottom margins of print layout. |
| left | Specifies the left margins of print layout. |

```
<span style="font-weight:bold">Example</span>  
'js  
<div id="reportviewer"></div>  
<script>  
$("#reportviewer").boldReportViewer(  
{  
pageSettings: {  
margins: { top: 0.5, right: 0.25, bottom: 0.25, left: 0.25 }  
}  
});  
</script>  
'
```

parameters `array`

Properties

```
<h3 class="doc-prop-wrapper" id="labels" data-Path="labels-labels">  
<a href="#labels" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>labels</span>  
<span class="doc-prop-type"> array  
</span>  
</h3>  
Gets or sets the parameter labels.  
Defaults to null  
<span style="font-weight:bold">Example</span>  
'js  
<div id="reportviewer"></div>  
<script>  
$("#reportviewer").boldReportViewer({  
parameters: [{  
name:"Vehicle",  
labels:["Motor Bikes"],  
prompt:"Please select the color",  
values:["Red","Green","Blue","Yellow","Black"],  
nullable:false  
}]  
});  
</script>  
'  
  
<h3 class="doc-prop-wrapper" id="name" data-Path="name-name">  
<a href="#name" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>name</span>  
<span class="doc-prop-type"> string
```

```
</span>
```

```
</h3>
```

Gets or sets the name of the parameter.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
```

```
`js
```

```
<div id="reportviewer"></div>
```

```
<script>
```

```
$("#reportviewer").boldReportViewer({  
parameters: [{  
name:"Vehicle",  
labels:["Motor Bikes"],  
prompt:"Please select the color",  
values:["Red","Green","Blue","Yellow","Black"],  
nullable:false  
}]  
});  
</script>  
,
```

```
<h3 class="doc-prop-wrapper" id="nullable" data-Path="nullable-nullable">  
<a href="#nullable" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>nullable</span>  
<span class="doc-prop-type"> boolean  
</span>
```

```
</h3>
```

Gets or sets whether the parameter allows nullable value or not.

Defaults to *false*

```
<span style="font-weight:bold">Example</span>
```

parameters `array`

Properties

```
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({
parameters: [{{
name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
}}]
});
</script>
` 

<h3 class="doc-prop-wrapper" id="prompt" data-Path="prompt-prompt">
<a href="#prompt" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>prompt</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the prompt message associated with the specified parameter.

Defaults to *empty*

Example

```
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({
parameters: [{{

```

parameters `array` Properties

```
name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the Color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
}]
});
</script>
`  
  

<h3 class="doc-prop-wrapper" id="values" data-Path="values-values">
<a href="#values" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>values</span>
<span class="doc-prop-type"> array
</span>
</h3>
Gets or sets the parameter values.  

Defaults to []
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer({
parameters: [{  

name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
```

```
exportSettings Properties
```

```
}]  
});  
</script>  
'
```

exportSettings

Properties

```
<h3 class="doc-prop-wrapper" id="exportoptions" data-Path="exportoptions-exportOptions">  
  <a href="#exportoptions" aria-hidden="true" class="anchor">  
    <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
      <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
    </svg>  
  </a><span class='doc-prop-name'>exportOptions</span>  
  <span class="doc-prop-type"> enum  
  </span>  
</h3>  
<ts name = "ej.ReportViewer.ExportOptions" style = "display:none"></ts>
```

Specifies the export formats.

| Name | Description |
|-------------|--------------------------------------------------------------------------------|
| All | Specifies the All property in ExportOptions to get all available options. |
| Pdf | Specifies the Pdf property in ExportOptions to get Pdf option. |
| Word | Specifies the Word property in ExportOptions to get Word option. |
| Excel | Specifies the Excel property in ExportOptions to get Excel option. |
| Html | Specifies the Html property in ExportOptions to get Html option. |
| PPT | Specifies the PPT property in ExportOptions to get PPT option. |
| CSV | Specifies the CSV property in ExportOptions to get CSV option. |
| CustomItems | Specifies the customItems property in ExportOptions to get customItems option. |

Defaults to *ej.ReportViewer.ExportOptions.All*

Example

'js

```

<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  exportSettings:{ exportOptions: ej.ReportViewer.ExportOptions.Html |
ej.ReportViewer.ExportOptions.Pdf }
});
</script>
` 

<h3 class="doc-prop-wrapper" id="excelformat" data-Path="excelformat-excelFormat">
<a href="#excelformat" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>excelFormat</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.ExcelFormats" style = "display:none"></ts>

```

Specifies the excel export format.

| Name | Description |
|---------------|---------------------------------------------------------------------------------------------------|
| Excel97to2003 | Specifies the Excel97to2003 property in ExcelFormats to get specified version of exported format. |
| Excel2007 | Specifies the Excel2007 property in ExcelFormats to get specified version of exported format. |
| Excel2010 | Specifies the Excel2010 property in ExcelFormats to get specified version of exported format. |
| Excel2013 | Specifies the Excel2013 property in ExcelFormats to get specified version of exported format. |

Defaults to *ej.ReportViewer.ExcelFormats.Excel97to2003*

Example

```

`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  exportSettings:{ excelFormat: ej.ReportViewer.ExcelFormats.Excel97to2003}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="wordformat" data-Path="wordformat-wordFormat">
<a href="#wordformat" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>wordFormat</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.WordFormats" style = "display:none"></ts>

```

Specifies the word export format.

| Name | Description |
|----------|---------------------------------------------------------------------------------------------|
| Doc | Specifies the Doc property in WordFormats to get specified version of exported format. |
| Dot | Specifies the Dot property in WordFormats to get specified version of exported format. |
| DOCX | Specifies the DOCX property in WordFormats to get specified version of exported format. |
| Word2007 | Specifies the Word2007 property in WordFormats to get specified version of exported format. |
| Word2010 | Specifies the Word2010 property in WordFormats to get specified version of exported format. |

| Name | Description |
|--------------|-------------------------------------------------------------------------------------------------|
| Word2013 | Specifies the Word2013 property in WordFormats to get specified version of exported format. |
| Word2007Dotx | Specifies the Word2007Dotx property in WordFormats to get specified version of exported format. |
| Word2010Dotx | Specifies the Word2010Dotx property in WordFormats to get specified version of exported format. |
| Word2013Dotx | Specifies the Word2013Dotx property in WordFormats to get specified version of exported format. |
| Word2007Docm | Specifies the Word2007Docm property in WordFormats to get specified version of exported format. |
| Word2010Docm | Specifies the Word2010Docm property in WordFormats to get specified version of exported format. |
| Word2013Docm | Specifies the Word2013Docm property in WordFormats to get specified version of exported format. |
| Word2007Dotm | Specifies the Word2007Dotm property in WordFormats to get specified version of exported format. |
| Word2010Dotm | Specifies the Word2010Dotm property in WordFormats to get specified version of exported format. |
| Word2013Dotm | Specifies the Word2013Dotm property in WordFormats to get specified version of exported format. |
| RTF | Specifies the RTF property in WordFormats to get specified version of exported format. |
| Txt | Specifies the Txt property in WordFormats to get specified version of exported format. |
| EPUB | Specifies the EPUB property in WordFormats to get specified version of exported format. |
| HTML | Specifies the HTML property in WordFormats to get specified version of exported format. |
| XML | Specifies the XML property in WordFormats to get specified version of exported format. |
| Automatic | Specifies the Automatic property in WordFormats to get specified version of exported format. |

Defaults to `ej.ReportViewer.WordFormats.Doc`

Example

```
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
    exportSettings:{ wordFormat: ej.ReportViewer.WordFormats.Doc}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="customitems" data-Path="customitems-customItems">
<a href="#customitems" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>customItems</span>
<span class="doc-prop-type"> array
</span>
</h3>
Add the custom icon item to the export options.
Defaults to empty
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
    exportSettings: {
        customItems: [{ 
            index: 2,
            cssClass: "",
            value: 'exportCustom',

```

toolbarSettings

Properties

```
},  
{  
index: 4,  
cssClass: "",  
value: 'exportCustom3',  
}  
},  
});  
</script>  
'
```

toolbarSettings

Properties

```
<h3 class="doc-prop-wrapper" id="click" data-Path="click-click">  
<a href="#click" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>click</span>  
<span class="doc-prop-type"> string  
</span>  
</h3>
```

Fires when user click on toolbar item in the toolbar.

Defaults to *empty*

Example

`js

```
<div id="reportviewer"></div>
```

```
<script>
```

```
$("#reportviewer").boldReportViewer(
```

```
{
```

```
toolbarSettings:{click: "onToolbarClick"}
```

```
});
```

```

</script>
` 

<h3 class="doc-prop-wrapper" id="items" data-Path="items-items">
<a href="#items" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>items</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.ToolbarItems" style = "display:none"></ts>

```

Specifies the toolbar items.

| Name | Description |
|----------------|------------------------------------------------------------------------------|
| Print | Specifies the Print as value in ToolbarItems to get specified item. |
| Refresh | Specifies the Refresh as value in ToolbarItems to get specified item. |
| Stop | Specifies the Stop as value in ToolbarItems to get specified item. |
| Zoom | Specifies the Zoom as value in ToolbarItems to get specified item. |
| FitToPage | Specifies the FitToPage as value in ToolbarItems to get specified item. |
| Export | Specifies the Export as value in ToolbarItems to get specified item. |
| PageNavigation | Specifies the PageNavigation as value in ToolbarItems to get specified item. |
| Parameters | Specifies the Parameters as value in ToolbarItems to get specified item. |
| PrintLayout | Specifies the PrintLayout as value in ToolbarItems to get specified item. |
| PageSetup | Specifies the PageSetup as value in ToolbarItems to get specified item. |

Defaults to *null*

```

<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>

```

```
$("#reportviewer").boldReportViewer(  
{  
toolbarSettings:{ items: ej.ReportViewer.ToolbarItems.All }  
});  
</script>  
  
<h3 class="doc-prop-wrapper" id="showtoolbar" data-Path="showtoolbar-showToolbar">  
<a href="#showtoolbar" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>showToolbar</span>  
<span class="doc-prop-type"> boolean  
</span>  
</h3>  
Shows or hides the toolbar.  
Defaults to true  
<span style="font-weight:bold">Example</span>  
'js  
<div id="reportviewer"></div>  
<script>  
$("#reportviewer").boldReportViewer(  
{  
toolbarSettings:{ showToolbar: true }  
});  
</script>  
  
<h3 class="doc-prop-wrapper" id="showtooltip" data-Path="showtooltip-showTooltip">  
<a href="#showtooltip" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>showTooltip</span>
```

```
<span class="doc-prop-type"> boolean
```

```
</span>
```

```
</h3>
```

Shows or hides the tooltip of toolbar items.

Defaults to *true*

```
<span style="font-weight:bold">Example</span>
```

```
`js
```

```
<div id="reportviewer"></div>
```

```
<script>
```

```
$("#reportviewer").boldReportViewer(
```

```
{
```

```
toolbarSettings:{ showTooltip: true }
```

```
});
```

```
</script>
```

```
'
```

```
<h3 class="doc-prop-wrapper" id="templateid" data-Path="templateid-templateId">
```

```
<a href="#templateid" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>templateId</span>
```

```
<span class="doc-prop-type"> string
```

```
</span>
```

```
</h3>
```

Specifies the toolbar template ID.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
toolbarSettings:{ templateId: "customtoolbarId" }
});
</script>
` 

<h3 class="doc-prop-wrapper" id="customitems" data-Path="customitems-customItems">
<a href="#customitems" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>customItems</span>
<span class="doc-prop-type"> array
</span>
</h3>
Add the custom icon item to the toolbar.
Defaults to empty
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
toolbarSettings:{
customItems: [{groupIndex: 1, index: 1,

```

```
itemType: 'Default',
cssClass: "e-icon e-mail e-reportviewer-icon CustomItem",
tooltip: { header: 'CustomItem', content: 'toolbaritems', value: 'CustomItem' },
}]
}
});
};

</script>
`  
  

<h3 class="doc-prop-wrapper" id="customgroups" data-Path="customgroups-customGroups">
<a href="#customgroups" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>customGroups</span>
<span class="doc-prop-type"> array
</span>
</h3>
Add the custom icon groups to the toolbar.

Defaults to empty
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  toolbarSettings:{
    customGroups: [
      {
        items: [
          {
            itemType: 'Default',
            cssClass: "e-icon e-mail e-reportviewer-icon CustomGroup",
            tooltip: { header: 'CustomGroup', content: 'toolbargroups', value: 'CustomGroup' },

```

parameterSettings

Properties

```
},  
{  
itemType: 'Default',  
cssClass: "e-icon e-mail e-reportviewer-icon subCustomGroup",  
tooltip: { header: 'subCustomGroup', content: 'subtoolbargroups', value: 'subCustomGroup' },  
}  
},  
groupIndex: 3  
}],  
}  
});  
</script>  
,
```

parameterSettings

Properties

```
<h3 class="doc-prop-wrapper" id="delimiterchar" data-Path="delimiterchar-delimiterChar">  
<a href="#delimiterchar" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>delimiterChar</span>  
<span class="doc-prop-type"> string  
</span>  
</h3>
```

Sets the separator when the multiSelectMode with delimiter option or checkbox is enabled with the dropdown. When you enter the delimiter value, the texts after the delimiter are considered as a separate word or query. The delimiter string is a single character and must be a symbol. Mostly, the delimiter symbol is used as comma (,) or semi-colon (;) or any other special character.

Defaults to ''

```
<span style="font-weight:bold">Example</span>  
'js  
<div id="reportviewer"></div>  
<script>
```

```
$( "#reportviewer" ).boldReportViewer(  
{  
    parameterSettings:{ delimiterChar: "," }  
});  
</script>  
  
<h3 class="doc-prop-wrapper" id="popupheight" data-Path="popupheight-popupHeight">  
    <a href="#popupheight" aria-hidden="true" class="anchor">  
        <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
            <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
        </svg>  
    </a><span class='doc-prop-name'>popupHeight</span>  
    <span class="doc-prop-type"> string  
    </span>  
</h3>  
Specifies the height of the combobox parameter popup list. By default, the popup height value is 152px.  
Defaults to 152px  
Example  
'js  
<div id="reportviewer"></div>  
<script>  
    $( "#reportviewer" ).boldReportViewer(  
    {  
        parameterSettings:{ popupHeight: "200px" }  
    });  
</script>  
  
<h3 class="doc-prop-wrapper" id="popupwidth" data-Path="popupwidth-popupWidth">  
    <a href="#popupwidth" aria-hidden="true" class="anchor">  
        <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>popupWidth</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Specifies the width of the combobox parameter popup list. By default, the popup width sets based on the width of the component.

Defaults to '*auto*'

```
<span style="font-weight:bold">Example</span>
```

```
'js
```

```
<div id="reportviewer"></div>
```

```
<script>
```

```
$("#reportviewer").boldReportViewer(
```

```
{
```

```
parameterSettings:{ popupWidth: "150px" }
```

```
});
```

```
</script>
```

```
'
```

```
<h3 class="doc-prop-wrapper" id="itemwidth" data-Path="itemwidth-itemWidth">
<a href="#itemwidth" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>itemWidth</span>
```

```
<span class="doc-prop-type"> string
</span>
```

```
</h3>
```

Specifies the width of the parameter item. By default, the item width value is set as **185px**.

Defaults to *185px*

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
    parameterSettings:{ itemWidth: "250px" }
});
</script>
`
```

```
<h3 class="doc-prop-wrapper" id="labelwidth" data-Path="labelwidth-labelWidth">
<a href="#labelwidth" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0 -.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>labelWidth</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Specifies the width of the parameter label. By default, the parameter label width value is set as *110px*.

Defaults to *110px*

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
    parameterSettings:{ labelWidth: "auto" }
});
</script>
```

`

```
<h3 class="doc-prop-wrapper" id="mindatetime" data-Path="mindatetime-minDateTime">
<a href="#mindatetime" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>minDateTime</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Specifies the minimum date in the date parameter that the user can select. By default, the minDateTime value is set as null

Defaults to null

```
<span style="font-weight:bold">Example</span>
```

`js

```
<div id="reportviewer"></div>
<script>
$("#reportviewer").boldReportViewer(
{
  parameterSettings:{ minDateTime: "3/1/2003" }
});
```

</script>

`

```
<h3 class="doc-prop-wrapper" id="maxdatetime" data-Path="maxdatetime-maxDateTime">
<a href="#maxdatetime" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>maxDateTime</span>
```

```
<span class="doc-prop-type"> string  
</span>  
</h3>
```

Specifies the maximum date in the date parameter that the user can select. By default, the **maxDateTime** value is set as **null**

Defaults to *null*

Example

`js

```
<div id="reportviewer"></div>
```

```
<script>
```

```
$("#reportviewer").boldReportViewer(
```

```
{
```

```
parameterSettings:{ maxDateTime: "4/30/2003" }
```

```
});
```

```
</script>
```

```
\
```

```
<h3 class="doc-prop-wrapper" id="hideparameterblock" data-Path="hideparameterblock-hideParameterBlock">
```

```
<a href="#hideparameterblock" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>hideParameterBlock</span>
```

```
<span class="doc-prop-type"> boolean
```

```
</span>
```

```
</h3>
```

Show or hide the parameter block on report initial rendering.

Defaults to *false*

Example

`js

```
<div id="reportviewer"></div>
```

```
<script>
```

dataSources `array` Properties

```
$("#reportviewer").boldReportViewer(  
{  
parameterSettings:{ hideParameterBlock: true }  
});  
</script>  
'
```

dataSources `array`

Properties

```
<h3 class="doc-prop-wrapper" id="name" data-Path="name-name">  
<a href="#name" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>name</span>  
<span class="doc-prop-type"> string  
</span>
```

Gets or sets the name of the data source.

Defaults to *empty*

Example

`js

```
<div id="reportviewer"></div>
```

<script>

```
$("#reportviewer").boldReportViewer({ dataSources: [name:"Menu Items",  
values:[{ OrderId: "21D60", FoodName: "Burger", Price: 20, Category: "Veg" },  
{ OrderId: "21D61", FoodName: "Pizza", Price: 25, Category: "Non-Veg" },  
{ OrderId: "21D63", FoodName: "Sandwiches", Price: 30, Category: "Non-Veg" },  
{ OrderId: "21D65", FoodName: "Chicken Drum Sticks", Price: 23, Category: "Non-Veg" },  
{ OrderId: "21D64", FoodName: "Fulka", Price: 15, Category: "Veg" }]]  
});  
</script>
```

```

`  

<h3 class="doc-prop-wrapper" id="value" data-Path="value-value">  

<a href="#value" aria-hidden="true" class="anchor">  

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  

.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-  

1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  

3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  

</svg>  

</a><span class='doc-prop-name'>value</span>  

<span class="doc-prop-type"> array  

</span>  

</h3>  

Gets or sets the values of data source.  

Defaults to []  

<span style="font-weight:bold">Example</span>  

`js  

<div id="reportviewer"></div>  

<script>  

$( "#reportviewer" ).boldReportViewer({ dataSources: [name:"Menu Items",  

value:[{ OrderId: "21D60", FoodName: "Burger", Price: 20, Category: "Veg" },  

{ OrderId: "21D61", FoodName: "Pizza", Price: 25, Category: "Non-Veg" },  

{ OrderId: "21D63", FoodName: "Sandwiches", Price: 30, Category: "Non-Veg" },  

{ OrderId: "21D65", FoodName: "Chicken Drum Sticks", Price: 23, Category: "Non-Veg" },  

{ OrderId: "21D64", FoodName: "Fulka", Price: 15, Category: "Veg" }]]  

});  

</script>  

`
```

Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports JavaScript Report Viewer.

[How can improve the performance and handle the large amounts of data with Report Viewer?](#)

[Is Report Viewer compatible with latest version of JQuery library?](#)

How can improve the performance and handle the large amounts of data with Report Viewer Properties

[Is the back action from drillthrough report will load the report again with Report Viewer?](#)

[Is possible to change the culture of the date time parameter?](#)

[Is it possible to hide report parameters?](#)

[Is it possible to hide the export options in Report Viewer?](#)

[Is it possible to load reports providing parameter values at runtime?](#)

How can improve the performance and handle the large amounts of data with Report Viewer

You have to use the **EnableVirtualEvaluation** and **DisablePageSplitting** with ReportViewer to handle the larger amount data. This will helpful to process the report with less memory footprint.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.EnableVirtualEvaluation = true;
    reportOption.ReportModel.DisablePageSplitting = true;
}
```

To improve the loading performance, you are requesting to set **CanGrow** true for necessary textbox alone in report. Because, if you set the **CanGrow** for textboxes and report having larger amount, then viewer will takes time, measure, height, and width for each textbox cells based on amount of data.

[Is Report Viewer compatible with latest version of JQuery library](#)

Yes, Report Viewer is compatible with latest version of JQuery library. You can make use of the latest JQuery library based on your need.

[Is the back action from drillthrough report will load the report again with Report Viewer](#)

No, Report Viewer will not load or process the report again when back from drillthrough report.

[Is possible to change the culture of the date time parameter](#)

Yes, we can change the culture of the date time parameter for Report Viewer by changing the locale. You can make use the following reference to change the locale of Report Viewer.

[ReportViewer Localization](#)

In Report Viewer, we could not change the culture of specific parameter or UI. So, we have to achieve the requirements only by changing the culture.

Is it possible to hide the parameters in Report Viewer

Yes, it is possible to hide the parameters in Report Viewer. On hiding the parameters, the users will not be able to have the parameter block in the Report Viewer. Refer to this [Hide parameter block and toolbar items](#) section.

Is it possible to hide the export options in Report Viewer

Yes, it is possible to hide the export options in Report Viewer. The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. Refer to this [Decide or Hide the export options](#) section.

Is it possible to load reports providing parameter values at runtime

Yes, it is possible to load reports with application inputs as parameters in Report Viewer. You need to provide the input values to the Report Viewer from client side at runtime using the `parameters` property, which accepts JSON array values. Refer to this [Set parameter at client](#) section.

Reporting tools for JavaScript

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

Key features

Data Sources --- Supports connection to major data providers such as

Microsoft SQL Server, Oracle, OLEDB and ODBC for exploring data and design reports with a wide range of data sources.

User friendly Environment --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

Report items --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

Report Parameter --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

Expression --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

Add Web Report Designer to a javascript application

This section explains the steps required to add Web Report Designer to a javascript application.

HTML file creation

Create a basic HTML file and place it in a separate folder. Add the below code snippet,

```
'html
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
<title>Report Designer first HTML page</title>
</head>
<body>
</body>
</html>
`
```

Refer Scripts and CSS

Directly refer the required scripts and style sheets that are mandatorily required to render the web Report Designer, from **CDN** links.

Refer to the [Dependencies](#) to learn more details about web Report Designer dependent scripts and style sheets links.

You can use the following code in the <head> tag of the HTML page as in the following order.

```
'html
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css"
rel="stylesheet" />
<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css"
rel="stylesheet" />
<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css"
rel="stylesheet" />
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
<script src="https://cdn.boldreports.com/external/jsrender.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js"
type="text/javascript"></script>
<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
```

```

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-
widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"
type="text/javascript"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js"
type="text/javascript"></script>
`
```

Whether you want to get the scripts and style sheets as local, then install the **BoldReports.Javascript** NuGet package in your application.

Initialize Report Designer

Add the `<div>` element within the `<body>` section, which acts as a container for `boldReportDesigner` widget to render and then initialize the `boldReportDesigner` widget within the script section as shown below,

```

`html
<!-- Creating a div tag which will act as a container for boldReportDesigner widget.-->
<div style="height: 600px; width: 950px; min-height: 400px;" id="designer"></div>
<script type="text/javascript">
$(function () {
    $("#designer").boldReportDesigner();
});
</script>
`
```

Create Web API service

The web Report Designer requires a Web API service to process the data and file actions, so you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

This tutorial uses the same Web API service application created in [Create ASP.NET Web API Service](#) tutorial.

Set service URL

Set the Web API controller name to the `serviceUrl` property of the web Report Designer.

Add the following code in the `<body>` tag of the web Report Designer HTML page.

```
'html
<!-- Creating a div tag which will act as a container for boldReportDesigner widget.-->
<div style="height: 600px; width: 950px; min-height: 400px;" id="designer"></div>
<script type="text/javascript">
$(function () {
    $("#designer").boldReportDesigner({
        serviceUrl: "/api/ReportingAPI"
    });
});
</script>
`
```

Run the application

Build and run the javascript application. Preview and edit the result using the following.

```
{% tab demoPath="javascript/report-designer/getting-started/initialize-report-designer"
files="index.html,index.js,ReportingAPIController.cs" %}

{% tabItem header="index.html" %}

`html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="designer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
`


{% endtabItem %}
```

```
{% tabItem header="index.js" %}  
`javascript  
$(function () {  
    $("#designer").boldReportDesigner({  
        serviceUrl: "https://demos.boldreports.com/services/api/ReportingAPI"  
    });  
});  
  
{% endtabItem %}  
{% tabItem header="ReportingAPIController.cs" %}  
`csharp  
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Net;  
using System.Net.Http;  
using System.Web.Http;  
using BoldReports.Web;  
using BoldReports.Web.ReportViewer;  
using BoldReports.Web.ReportDesigner;  
namespace ReportDesignerAPIService  
{  
    [System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]  
    public class ReportingAPIController : ApiController, IReportDesignerController  
    {  
        /// <summary>  
        /// Get the path of specific file  
        /// </summary>  
        /// <param name="itemName">Name of the file to get the full path</param>  
        /// <param name="key">Name of the file to get the full path</param>  
        /// <returns>Returns the full path of file</returns>  
        private string GetFilePath(string itemName, string key)  
        {
```

```
string targetFolder = HttpContext.Current.Server.MapPath("~/");
targetFolder += "Cache";
if (!Directory.Exists(targetFolder))
{
    Directory.CreateDirectory(targetFolder);
}
if (!Directory.Exists(targetFolder + "\\\" + key))
{
    Directory.CreateDirectory(targetFolder + "\\\" + key);
}
return targetFolder + "\\\" + key + "\\\" + itemName;
}

/// <summary>
/// Action (HttpGet) method for getting resource for report images.
/// </summary>
/// <param name="key">The unique key for request identification.</param>
/// <param name="image">The name of requested image.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetImage")]
[AcceptVerbs("GET")]
public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}

/// <summary>
/// Action (HttpPost) method for posting the request for designer actions.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing designer request.</param>
/// <returns>The object data.</returns>
public object PostDesignerAction(Dictionary<string, object> jsonData)
{
    //Processes the designer request and returns the result.
    return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);
}
```

```
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = Path.GetFileName(itemData.PostedFile.FileName);
        }
        itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));
    }
    return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

/// <summary>
/// Action (HttpPost) method for uploaded file actions.
/// </summary>

public void UploadReportAction()
{
    //Processes the designer file upload request's.
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}
```

```
/// <summary>
/// Action (HttpGet) method for getting resource to report.
/// </summary>
/// <param name="key">The unique key to get the required resource.</param>
/// <param name="resourcetype">The type of the requested resource.</param>
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]
public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

/// <summary>
/// Report Initialization method that is triggered when report begin processed.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        //Gauge component scripts
        resourcesPath + @"\bold-reports\common\ej2-base.min.js",
        resourcesPath + @"\bold-reports\common\ej2-data.min.js",
        resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
        resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
        resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
    }
}
```

```
resourcesPath + @"\\bold-reports\\data-visualization\\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.chart.min.js",
resourcesPath + @"\\bold-reports\\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\\jquery-1.7.1.min.js"
};
}

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOption)
{
//You can update report options here
}
/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
//Processes the report request and returns the result.

return ReportHelper.ProcessReport(jsonData, this as IReportController);
}
}
`
```

```
{% endtabItem %}  

{% endtab %}
```

Dependencies

The ReportDesigner have the following list of internal and external dependencies to render the component.

Scripts

Internal dependencies

| Name | Details | CDN link |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| bold.reports.common.min.js | Common script for reporting widgets. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js |
| bold.reports.widgets.min.js | Supports Syncfusion widgets to render in HTML5 format and it contains the dependent Syncfusion controls that is common for both report designer and viewer. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js |
| bold.report-designer-widgets.min.js | Supports Syncfusion widgets to | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-widgets.min.js Unsecured |

| | | |
|-----------------|------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | render in HTML5 format and it contains the dependent Syncfusion controls for report designer | Link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-widgets.min.js |
| ej.chart.min.js | Renders the chart gauge item. Add this script only if your report contains the chart gauge report item. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js |
| ej2-base.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js |

| | | |
|------------------------|--------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej2-data.min.js | <p>Renders the gauge item. Add this script only if your report contains the gauge report item.</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js Unsecured Link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js</p> |
| ej2-pdf-export.min.js | <p>Renders the gauge item. Add this script only if your report contains the gauge report item.</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js Unsecured Link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js</p> |
| ej2-svg-base.min.js | <p>Renders the gauge item. Add this script only if your report contains the gauge report item.</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js Unsecured Link:https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js</p> |
| ej2-lineargauge.min.js | <p>Renders the linear gauge</p> | <p>Secured link:https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js</p> |

| | | |
|---------------------------|------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | item. Add this script only if your report contains the linear gauge report item. | Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js |
| ej2-circulargauge.min.js | Renders the circular gauge item. Add this script only if your report contains the circular gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js |
| ej.map.min.js | Renders the map item. Add this script only if your report contains the map report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js |
| bold.report-viewer.min.js | Used to preview the reports in report | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js |

| | | |
|-----------------------------|------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | designer. . | |
| bold.report-designer.min.js | Used to render the Syncfusion JavaScript Report Designer widget. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js |

External dependencies

| Name | Details | CDN link |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| jQuery 1.7.1 and later versions | Common jQuery script to render the Syncfusion JavaScript Reporting widgets | Secured Link: https://cdn.boldreports.com/external/jquery-1.10.2.min.js Unsecured Link: http://cdn.boldreports.com/external/jquery-1.10.2.min.js |
| jsrender | The script to render the templates in the browser. | Secured Link: https://cdn.boldreports.com/external/jsrender.min.js Unsecured Link: http://cdn.boldreports.com/external/jsrender.min.js |
| Codemirror | Report Designer requires the code mirror scripts to edit the SQL queries and Visual Basic code functions with syntax highlighter. | Secured Link: https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js Unsecured Link: http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js |

| | | |
|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js |
|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Styles

Internal dependencies

| Name | Details |
|-----------------------------|-------------------------------------------------------------------------|
| bold.reports.all.min.css | It contains the styles and css references of dependent components. |
| bold.reportdesigner.min.css | It contains the styles and css references of Report Designer component. |

The CDN links for all supported themes are provided in the below table. Refer the following syntax:

[https://cdn.boldreports.com/\[version\]/content/\[theme-name\]/\[fileName\]](https://cdn.boldreports.com/[version]/content/[theme-name]/[fileName])

| Name | Details | CDN link |
|-----------------|-----------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Material | Includes the CSS properties defined for the Syncfusion JavaScript Reporting component in material theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css |
| Bootstrap-theme | Includes the CSS properties defined for the Syncfusion JavaScript Reporting component in bootstrap theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reportdesigner.min.css |
| Office-365 | Includes the CSS properties defined for the | Secured Link: https://cdn.boldreports.com/2.2.32/content/office-365/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/office-365/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/office-365/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/office-365/bold.reportdesigner.min.css |

| | | |
|------------------|------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | Syncfusion JavaScript Reporting component in office-365 theme. | 365/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/office-365/bold.reportdesigner.min.css |
| High-contrast-01 | Includes the CSS properties defined for the Syncfusion JavaScript Reporting component in high-contrast-01 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reportdesigner.min.css |
| High-contrast-02 | Includes the CSS properties defined for the Syncfusion JavaScript Reporting component in high-contrast-02 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reportdesigner.min.css |

All the provided CDN links can be accessed either through `http` or `https`.

External dependencies

| Name | Details | CDN link |
|------------|----------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Codemirror | Report Designer requires the code mirror styles to edit the SQL queries and Visual Basic code functions with syntax highlighter. | Secured Link: https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css Unsecured Link: http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css |

| | | |
|--|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | codemirror.min.css show- hint.min.css | http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css |
|--|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Localization

Localization is the process of customizing the application for a given culture and locale - involving much more than the simple translation of text. In web report designer, localized strings appropriate to each culture are stored in separate files and loaded according to the UI culture setting.

By default report designer display strings in US English(en-US).

To render the static text with specific culture, refer the following corresponding culture script files and set culture name to the **locale** property of the Report Designer.

```
<ul>
<li> ej.localetexts.fr-FR.min.js </li>
<li> ej.culture.fr-FR.min.js </li>
</ul>
```

Refer the **ej.localetexts.fr-FR.min.js** and **ej.culture.fr-FR.min.js** scripts from cdn using the following code in **<head>** tag of the Report Designer HTML page.

```
'html
<script src="http://cdn.boldreports.com/2.2.32/scripts/l10n/reportdesigner/ej.localetexts.fr-FR.min.js"></script>
<script src="http://cdn.boldreports.com/2.2.32/scripts/i18n/ej.culture.fr-FR.min.js"></script>
'
```

Whether you want to get the localization script as local then install the **BoldReports.Global** NuGet package in your application.

```
'html
<script src="scripts/l10n/ej.localetexts.fr-FR.min.js"></script>
<script src="scripts/i18n/ej.culture.fr-FR.min.js"></script>
'
```

You can edit and preview the Report Designer localization using the following.

```
{% tab demoPath="javascript/report-designer/localization" files="index.html,index.js" %}
{% tabItem header="index.html" %}
'html
<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
```

```
<div id="designer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>
` 

{%- endtabItem %}

{%- tabItem header="index.js" %}

`javascript
$(function () {
    $("#designer").boldReportDesigner({
        serviceUrl: "https://demos.boldreports.com/services/api/ReportingAPI",
        locale: "fr-FR"
    });
});

` 

{%- endtabItem %}

{%- tabItem header="ReportingAPIController.cs" %}

`csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using BoldReports.Web;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
namespace ReportDesignerAPIService
{
    [System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
    public class ReportingAPIController : ApiController, IReportDesignerController
    {
        /// <summary>
        /// Get the path of specific file

```

```
/// </summary>
/// <param name="itemName">Name of the file to get the full path</param>
/// <param name="key">The unique key for report designer.</param>
/// <returns>Returns the full path of file</returns>
private string GetFilePath(string itemName, string key)
{
    string targetFolder = HttpContext.Current.Server.MapPath("~/");
    targetFolder += "Cache";
    if (!Directory.Exists(targetFolder))
    {
        Directory.CreateDirectory(targetFolder);
    }
    if (!Directory.Exists(targetFolder + "\\\" + key))
    {
        Directory.CreateDirectory(targetFolder + "\\\" + key);
    }
    return targetFolder + "\\\" + key + "\\\" + itemName;
}
/// <summary>
/// Action (HttpGet) method for getting resource for report images.
/// </summary>
/// <param name="key">The unique key for request identification.</param>
/// <param name="image">The name of requested image.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetImage")]
[AcceptVerbs("GET")]
public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}
/// <summary>
/// Action (HttpPost) method for posting the request for designer actions.
/// </summary>
```

```
/// <param name="jsonData">The JSON data posted for processing designer request.</param>
/// <returns>The object data.</returns>
public object PostDesignerAction(Dictionary<string, object> jsonData)
{
    //Processes the designer request and returns the result.
    return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = Path.GetFileName(itemData.PostedFile.FileName);
        }
        itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));
    }
    return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

/// <summary>
/// Action (HttpPost) method for uploaded file actions.

```

```
/// </summary>
public void UploadReportAction()
{
//Processes the designer file upload request's.

ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);

}
/// <summary>
/// Action (HttpGet) method for getting resource to report.
/// </summary>
/// <param name="key">The unique key to get the required resource.</param>
/// <param name="resourcetype">The type of the requested resource.</param>
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]

public object GetResource(string key, string resourcetype, bool isPrint)
{
return ReportHelper.GetResource(key, resourcetype, isPrint);
}

/// <summary>
/// Report Initialization method that is triggered when report begin processed.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
reportOption.ReportModel.ExportResources.Scripts = new List<string>
{
resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
//Chart component script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",

```

```
//Gauge component scripts
resourcesPath + @"\bold-reports\common\ej2-base.min.js",
resourcesPath + @"\bold-reports\common\ej2-data.min.js",
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
    resourcesPath + @"\jquery-1.7.1.min.js"
};

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
```

```
//Processes the report request and returns the result.  
return ReportHelper.ProcessReport(jsonData, this as IReportController);  
}  
}  
}  
`  
  
{% endtabItem %}  
{% endtab %}
```

Integrate the component with Report Server

You can integrate Report Designer with Report Server to create, edit, browse and publish reports using the Report Server built-in API service.

The Report Designer requires the `serviceAuthorizationToken` to perform the API actions with Bold Report Server. Create a token for the user by using the server RESTful API, the following code is used to generate the token.

```
`js  
<script type="text/javascript">  
$(function () {  
    var dataValue = "";  
    var apiRequest = new Object();  
    apiRequest.password = "demo";  
    apiRequest.userid = "guest@boldreports.com";  
    $.ajax({  
        type: "POST",  
        url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",  
        data: apiRequest,  
        success: function (data) {  
            dataValue = data.Token;  
            var token = JSON.parse(dataValue);  
            // Report Designer initialization.  
        }  
    });  
});  
</script>
```

Set the Bold Report Server built-in service URL to the `serviceUrl` property and assign the created token to `serviceAuthorizationToken` property. You can use the following complete code in your HTML page.

```
'js
<script type="text/javascript">
$(function () {
var dataValue = "";
var apiRequest = new Object();
apiRequest.password = "demo";
apiRequest.userid = "guest@boldreports.com";
$.ajax({
type: "POST",
url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
data: apiRequest,
success: function (data) {
dataValue = data.Token;
var token = JSON.parse(dataValue);
$("#designer").boldReportDesigner(
{
serviceUrl: "https://on-premise-demo.boldreports.com/reporting/reportservice/api/Designer",
serviceAuthorizationToken: token["tokentype"] + " " + token["accesstoken"],
ajaxBeforeLoad: "onAjaxRequest"
});
}
});
});
};

function onAjaxRequest(args) {
args.headers.push({
Key: 'serverurl', Value: 'https://on-premise-demo.boldreports.com/reporting/api/site/site1'
});
}
</script>
```

Run the application

Build and run the application. Preview and edit the result using the following.

```
{% tab demoPath="javascript/report-designer/server-integration" files="index.html,index.js" %}

{% tabItem header="index.html" %}

`html

<body style="overflow: hidden; position: static; margin: 0; padding: 0; height: 100%; width: 100%;">
<div id="designer" style="position: absolute; height: 100%; width: 100%;"></div>
<script src="index.js"></script>
</body>

`


{% endtabItem %}

{% tabItem header="index.js" %}

`javascript

var isSubmit = true;

$(function () {
    $(document.body).bind('submit', $.proxy(formSubmit, this));
    $(window).bind('beforeunload', $.proxy(windowUnload, this));
    var dataValue = "";
    var apiRequest = new Object();
    apiRequest.password = "demo";
    apiRequest.userid = "guest@boldreports.com";
    $.ajax({
        type: "POST",
        url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
        data: apiRequest,
        success: function (data) {
            dataValue = data.Token;
            var token = JSON.parse(dataValue);
            $("#designer").boldReportDesigner(
            {
                serviceUrl: "https://on-premise-demo.boldreports.com/reporting/reportservice/api/Designer",
                serviceAuthorizationToken: token["tokentype"] + " " + token["accesstoken"],
            }
        )
    })
})
```

```
ajaxBeforeLoad: "onAjaxRequest"
});
}
});
});

function formSubmit(args) {
isSubmit = false;
}

function windowUnload(args) {
var designer = $('#designer').data('boldReportDesigner');
if (designer && designer.hasReportChanges() && isSubmit) {
return 'Changes you made may not be saved';
}
isSubmit = true;
}

function onAjaxRequest(args) {
args.headers.push({
Key: 'serverurl', Value: 'https://on-premise-demo.boldreports.com/reporting/api/site/site1'
});
}
.

{
  endtabItem
}
{
  endtab
}
```

Designing Reports

The Bold Report Designer provides an end-user documentation that describes how to interact with the Report Designer's UI and design an interactive reports. Refer the following user guide sections to get start with Bold Report Designer.

Connecting your data

A report must have a data source and dataset to include data. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

[Manage data source](#)

[Manage data set](#)

Transform your data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)

[Formatting columns](#)

[Query filter](#)

[Data set parameter](#)

[Query parameter](#)

[Configure expression columns](#)

Working with report parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it.

[Add report parameter](#)

[Edit report parameter](#)

[Delete report parameter](#)

[Cascading parameter](#)

[Multi-value parameter](#)

Embed images into the report

Supports to embed local or database images into the report and image data is stored within the report definition. The embedded images in a report are listed in the **Image Manager panel**.

[Embed images into the report](#)

Interacting with report design surface

WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

[Interacting with report design surface](#)

Report items

Offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

[Configure report items](#)

Customize appearance

The Properties panel shows all the properties of the selected section, report items or the report itself to customize the appearance of the report.

[Customize appearance](#)

Report design settings

[Configure design settings](#)

Shape report data

[Filter data](#)

[Group data](#)[Sort data](#)[Format data](#)

Interactive features

[Hyperlink](#)[Drilldown](#)[Interactive sorting](#)

Working with Expressions

Expressions are used to specify criteria for retrieving and formatting data, creating calculated fields and calculating summaries, conditionally shaping data and changing a report control's appearance.

[Expressions builder](#)

Open report

[Open report](#)

Save report

[Save report](#)

Preview report

[Preview report](#)

Samples and demos

Browse and explore our ready-to-use the designer samples, online and offline demos.

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view our JavaScript Report Designer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view get our GitHub Report Designer demo samples.

Reporting Service application

Click [here](#) to view the GitHub source location of the reporting services that are used in our documentation.

Migrate Report Designer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Designer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Designer from Syncfusion Essential Studio release version to Bold Reports version of JavaScript Report Designer application:

Client-side migration

Scripts

| Old Scripts | New Scripts | Description |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej.web.all.min.js | bold.reports.common.min.js bold.reports.widgets.min.js bold.report-designer-widgets.min.js bold.report-designer-widgets.min.js ej.chart.min.js ej2-base.min.js ej2-data.min.js ej2-pdf-export.min.js ej2-svg-base.min.js ej2-lineargauge.min.js ej2-circulargauge.min.js ej.map.min.js bold.report-viewer.min.js | <p>We have removed the dependency scripts for report designer from existing ej.web.all.min.js script and provided the dependency scripts as below:</p> <p>bold.reports.common.min.js - Common script for reporting widgets.</p> <p>bold.reports.widgets.min.js - It contains the scripts of dependent controls that are common for both Report Designer and Report Viewer.</p> <p>bold.report-designer-widgets.min.js - It contains the scripts of Report Designer dependent controls.</p> <p>ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart report item.</p> <p>ej2-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-data.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-pdf-export.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p>ej2-svg-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> |

| | | |
|---------------------------------------|------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | <p>report item.</p> <p><code>ej2-lineargauge.min.js</code> - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p><code>ej2-circulargauge.min.js</code> - Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Previews the reports designed with Report Designer.</p> |
| <code>ej.reportdesigner.min.js</code> | <code>bold.report-designer.min.js</code> | Renamed the report designer script file and it used to render the Bold Report Designer widget. |

Styles

| Old Scripts | New Scripts | Description |
|---------------------------------|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ej.web.all.min.css</code> | <code>bold.reports.all.min.css</code> | We have removed the dependent controls styles for report designer from existing <code>ej.web.all.min.css</code> script and provided the dependent controls styles as <code>bold.reports.all.min.css</code> . |

Remove the old scripts and styles references from existing application, then add the new scripts and styles references based on above script name changes.

Server-side migration

Assemblies

| Purpose | Old Assemblies | New Assemblies | Description |
|-----------------|---------------------------------------------|----------------------------------|-----------------------------------------------------|
| Base Assemblies | <code>Syncfusion.EJ.ReportViewer.dll</code> | <code>BoldReports.Web.dll</code> | The <code>Syncfusion.EJ.ReportViewer.dll</code> and |

| | | |
|--|----------------------------------|----------------------------------------------------------------------------------------|
| | Syncfusion.EJ.ReportDesigner.dll | Syncfusion.EJ.ReportDesigner.dll assemblies have been combined as BoldReports.Web.dll. |
|--|----------------------------------|----------------------------------------------------------------------------------------|

Packages

| Purpose | Old Packages | New Packages |
|---------------------------|---------------------------------------------------------------|------------------------|
| Server Side Helper | Syncfusion.Web.ReportDesigner Syncfusion.Web.ReportViewer | BoldReports.Web |
| Script and Style Packages | Syncfusion.JavaScript.ReportDesigner Syncfusion.JavaScript | BoldReports.JavaScript |

Namespace changes

| Assembly Name | Old Namespace | New Namespace |
|---------------------|-----------------------------------|--------------------------------|
| BoldReports.Web.dll | Syncfusion.Reports.EJ | BoldReports.Web |
| | Syncfusion.EJ.ReportWriter | BoldReports.Writer |
| | Syncfusion.EJ.ReportViewer | BoldReports.Web.ReportViewer |
| | Syncfusion.EJ.ReportDesigner | BoldReports.Web.ReportDesigner |
| | Syncfusion.Reports.EJ.Data | BoldReports.Data |
| | Syncfusion.Reporting | BoldReports.Configuration |
| | Syncfusion.EJ.RDL.ServerProcessor | BoldReports.ServerProcessor |

Based on above assembly and namespace changes, modify the Report Designer **Web API Controller** in your application.

Control initialization

| Old code snippet | New code snippet |
|---------------------------------------------------|-----------------------------------------------------|
| <code>\$("#container").ejReportDesigner();</code> | <code>\$("#container").boldReportDesigner();</code> |

Report export configuration

The **BoldReports.Web** assembly can export the reports with data visualization components such as chart, gauge, and map, only if we configure the web scripts in Report Designer Web API controller. To configure the scripts in Web API controller, refer to the following steps:

Open the Report Designer Web API controller.

Configure the following scripts and styles in **OnInitReportOptions** on Web API controller:

jquery-1.7.1.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js - Exports the chart item. Add this script only if your report contains the chart report item.

ej2-base.min.js, **ej2-data.min.js**, **ej2-pdf-export.min.js**, **ej2-svg-base.min.js**, **ej2-lineargauge.min.js** and **ej2-circulargauge.min.js** - Exports the gauge item. Add this script only if your report contains the gauge report item.

ej.map.min.js - Exports the map item. Add this script only if your report contains the map report item.

bold.report-viewer.min.js

You can replace the **OnInitReportOptions** action in Report Designer Web API controller using below code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        //Gauge component scripts
        resourcesPath + @"\bold-reports\common\ej2-base.min.js",
        resourcesPath + @"\bold-reports\common\ej2-data.min.js",
        resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
        resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
        resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
        resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
        //Map component script
    }
}
```

```
resourcesPath + @"\\bold-reports\\data-visualization\\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.chart.min.js",
resourcesPath + @"\\bold-reports\\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\\jquery-1.7.1.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

NuGet Packages for JavaScript

Refer to the following steps to configure Bold Reporting tools NuGet Packages for JavaScript application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting tools NuGet packages are published to [Nuget.org](#). To configure the online packages use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button (green plus), and enter the following **Package Name** and **Package Source URL** and click **Update**.

Name: NuGet.org

Source: <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/javascript/report-designer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reporting tools NuGet packages are shipped into our Report Platform SDK build. To configure the packages from Bold Reports installed location use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button (green plus), and enter the following **Package Name** and **Package Source URL** and click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages.

![Offline NuGet Configure](/static/assets/javascript/report-designer/images/nuget-packages/LocalNuGetConfig.png)

The system drive will vary based on the installed location in your machine.

Installing NuGet Packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in the Visual Studio solution or project:

On the **Tools**, menu, **NuGet Package Manager | Manage NuGet Packages for Solution**

Alternatively, right-click on the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source and search the packages (**BoldReports.Web**) and Click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager** and then **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

install specified package in default project

```
Install-Package <Package Name>
```

install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

install specified package in default project

Upgrading NuGet packages

For example:

`cmd

[install specified package in default project](#)

Install-Package BoldReports.Web

[install specified package in default project with specified Package Source](#)

Install-Package BoldReports.Web -Source "https://api.nuget.org/v3/index.json"

[install specified package in specified project](#)

Install-Package BoldReports.Web -ProjectName BoldReportingApplication

\

[Upgrading NuGet packages](#)

Upgrade using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, **NuGet Package Manager | Manage NuGet Packages for Solution...**

Alternatively, right-click on project/solution in the Solution Explorer tab, and choose **Manage NuGet Packages**

Select the **Updates** tab to see the packages available for update from the desired package sources.

Select the required packages and the specific version from the dropdown, and click the **Update** button.

[Upgrade using Package Manager Console](#)

To update the installed Bold Reporting tools NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then **Package Manager Console**.

Run the following NuGet installation commands:

`cmd

[Update specific NuGet package in default project](#)

Update-Package <Package Name>

[Update all the packages in default project](#)

Update-Package

Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

\

For example:

`cmd

Update specified Bold Reporting NuGet package

Update-Package BoldReports.Web

Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportingApplication

\

Upgrade using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest NuGet CLI from [here](#).

To update the existing `nuget.exe` to latest version use the following command:

`cmd

`nuget update -self`

\

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting tools NuGet packages.

`cmd

update all NuGet packages from config file

`nuget update <configPath> [options]`

update all NuGet packages from specified Packages Source

Normal layout

update all NuGet packages from specified Packages Source

nuget update -Source <Source Location> [optional]

\

configPath is optional. This identifies the **package.config** or solutions file lists the packages utilized in the project.

For example:

`cmd

Update all NuGet packages from config file

nuget update "C:\Users\SyncfusionApplication\package.config"

Update all NuGet packages from specified Packages Source

nuget update -Source "https://api.nuget.org/v3/index.json"

\

Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

Responsive layout rendering of HTML5 JavaScript Report Designer

Report Designer will adaptively render itself with optimal user interfaces for tablet or desktop form factors. It has built-in responsive support, so that it will adjust automatically based on the browser viewport. Setting width to 100% is simply enough to make it responsive. This helps your application to scale elegantly on all form factors with ease.

Normal layout

The following output shows the normal layout rendering of Report Designer tool bar items.

![Rendering of Report Designer tool bar items in normal layout](/static/assets/javascript/report-designer/images/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Designer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/javascript/report-designer/images/responsive-layout.png)

Supported Browsers

This document provides the list of web browsers supported by the Web Report Designer

| Desktop Browsers | Version |
|-------------------|---------|
| ----- | ----- |
| Internet Explorer | 11.0+ |
| Microsoft Edge | Current |
| Firefox | 22+ |

|Google Chrome|17+|

|Opera|12+|

|Safari|5+|

Accessibility

The term Accessibility in general, refers to the process of designing a product that is easily accessible to all kinds of people irrespective of their disabilities.

Sometimes, the content and the functionalities used in the websites cannot be conveyed to the users who mainly relies on screen readers and one who cannot use the mouse. To overcome such circumstances and to address these accessibility challenges, **Bold Report Designer** provides built-in compliance with the WAI-ARIA specifications. WAI-ARIA compliance for the widgets have been achieved by providing proper keyboard navigation support as well as by defining the required ARIA attributes to the DOM elements. Web Accessibility Initiative (WAI-ARIA) will make use of the widget's attribute information (roles, states, and properties) and then convey the appropriate information to the persons with disabilities.

Javascript Report Designer Reporting Service

The Javascript Report Designer requires a Web API service to process the data and file actions. The following topics explains how to create reporting Web API service to create, edit, and browse reports.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

IReportDesignerController

The interface **IReportDesignerController** has the declaration of action methods that are defined in Web API Controller to retrieve data, save, edit and browse reports from the server. The IReportDesignerController has the following action methods declaration.

| Methods | Description |
|---------------------|------------------------------------------------------------------------------------|
| PostDesignerAction | Action (HttpPost) method for posting the request for designer actions. |
| UploadReportAction | Action (HttpPost) method for posted file actions. |
| GetImage | Action (HttpGet) method for getting resource for report images. |
| SetData | Write the resource into storage location. |
| GetData | Read the resource from storage location. |
| PostReportAction | Action (HttpPost) method for posting the request for report process. |
| GetResource | Action (HttpGet) method for getting resource for report. |
| OnInitReportOptions | Report initialization method that is triggered when report begins to be processed. |

| OnReportLoaded | Report loaded method that is triggered when report and sub report begin loading. |

[ReportDesignerHelper](#)

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

[ReportHelper](#)

The class **ReportHelper** contains helper methods that help process Post or Get request from the Report Viewer control and returns the response to the Report Viewer control. It has the following methods.

Methods | Description

GetResource | Returns the report resource for the requested key.

ProcessReport | Processes the report request and returns the result.

`csharp

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using BoldReports.Web;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
namespace ReportDesignerAPIService
{
    [System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
    public class ReportingAPIController : ApiController, IReportDesignerController
    {
        /// <summary>
        /// Action (HttpGet) method for getting resource for report images.
        /// </summary>
        /// <param name="key">The unique key for request identification.</param>
```

```
/// <param name="image">The name of requested image.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetImage")]
[AcceptVerbs("GET")]
public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}

/// <summary>
/// Action (HttpPost) method for posting the request for designer actions.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing designer request.</param>
/// <returns>The object data.</returns>
public object PostDesignerAction(Dictionary<string, object> jsonData)
{
    //Processes the designer request and returns the result.
    return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);
}

/// <summary>
/// File upload method that is call while upload a file.
/// </summary>
/// <param name="key">The unique key for report designer</param>
/// <param name="itemId">The unique key to get the required resource.</param>
/// <param name="itemData">Gets the resource data.</param>
/// <param name="errorMessage">Returns the error message, if the write action is failed.</param>
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    //You can implement the file save logic
}

/// <summary>
/// File download method that is call while downloading a file.
/// </summary>
/// <param name="key">The unique key for report designer</param>
```

```
/// <param name="itemId">The unique key to get the required resource.</param>
public ResourceInfo GetData(string key, string itemId)
{
    //You can implement the file download logic
}

/// <summary>
/// Action (HttpPost) method for uploaded file actions.
/// </summary>
public void UploadReportAction()
{
    //Processes the designer file upload request's.
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}

/// <summary>
/// Action (HttpGet) method for getting resource to report.
/// </summary>
/// <param name="key">The unique key to get the required resource.</param>
/// <param name="resourcetype">The type of the requested resource.</param>
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]
public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

/// <summary>
/// Report Initialization method that is triggered when report begin processed.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
```

```
//You can update report options here
}

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOption)
{
//You can update report options here
}
/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
//Processes the report request and returns the result.
return ReportHelper.ProcessReport(jsonData, this as IReportController);
}
}
}
`
```

Create ASP.NET Web API Service

In this section, you will learn how to create a Web API Service for Report Designer using the new ASP.NET Empty Web Application template.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![Creating a new ASP.NET Web Application Project](/static/assets/javascript/report-designer/report-service/web-api-images/aspnet-new-project.png)

Then choose the empty application in template and click OK.

![Choosing a empty template in new Project](/static/assets/javascript/report-designer/report-service/web-api-images/aspnet-empty-application.png)

List of dependency libraries

The Web API service configuration requires reporting server-side assembly references.

Right-click the project/solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages...**

![Open nuget manager](/static/assets/javascript/report-designer/report-service/web-api-images/add-dependencies-nuget.png)

Alternatively, click **Tools** in menu, then select **NuGet Package Manager | Manage NuGet Packages for Solution...**

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web** NuGet packages, and install them in your Web Forms application.

![Install reporting web package](/static/assets/javascript/report-designer/report-service/web-api-images/nuget-reporting-web.png)

BoldReports.Web package will install the following required dependencies for Report Designer into your application. Click OK.

Package | Purpose

BoldReports.Web | Builds the server-side implementations.

The above nuget package will automatically install below dependency packages.

Packages | Purpose

Syncfusion.Pdf.AspNet | Exports the report to a PDF.

Syncfusion.DocIO.AspNet | Exports the report to a Word.

Syncfusion.XlsIO.AspNet | Exports the report to an Excel.

Syncfusion.Presentation.AspNet | Exports the report to an PowerPoint.

Install the **Microsoft.AspNet.WebApi.Cors** from Nuget to enable the Cross-Origin Resource Sharing (CORS) for Web API.

```
`csharp
```

```
Install-Package Microsoft.AspNet.WebApi.Cors
```

```
\
```

This command installs the latest package and updates all dependencies, including the core Web API libraries. Use the -Version flag to target a specific version. Browser security prevents Report Designer making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Add Routing Information

The following steps guide you to configure the routing to include an action name in the URI.

Right-click the project in the solution explorer and select **Add > New item**.

In the Add New Item window, select **Global Application class** and name it as **Global.asax**, and then click Add.

![Adding Global.asax file](/static/assets/javascript/report-designer/report-service/web-api-images/add-global-application-class.png)

Open the code-behind file **Global.asax.cs** and add the following using statement.

```
'csharp
using System.Web.Http;
'
```

Then add the the following code to the **Application_Start** method to register CORS.

```
'csharp
protected void Application_Start(object sender, EventArgs e)
{
    System.Web.Http.GlobalConfiguration.Configuration.EnableCors();
}
'
```

For more information about CORS, see [Configure CORS for WebAPI](#)

Add the following code to the **Application_Start** method to register the Routing configuration:

```
'csharp
protected void Application_Start(object sender, EventArgs e)
{
    System.Web.Http.GlobalConfiguration.Configuration.EnableCors();
    System.Web.Http.GlobalConfiguration.Configuration.Routes.MapHttpRoute(
        name: "DefaultApi",
        routeTemplate: "api/{controller}/{action}/{id}",
    );
}
```

```
defaults: new { id = RouteParameter.Optional });
}
`
```

For more information about routing tables, see [Routing in ASP.NET Web API](#).

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **Web API Controller** class and name it as **ReportingAPIController**, and then click **Add**.

![Adding a new controller to the project](/static/assets/javascript/report-designer/report-service/web-api-images/add-web-api-controller.png)

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Configure Web API

The **IReportDesignerController** interface contains the required actions and helper methods declaration to process the designer file and data actions. The **ReportDesignerHelper** and **ReportHelper** class contains methods that help to process Post or Get request from the control and return the response.

| Methods | Description |
|--------------------------------------------------------------------------------------------------------------|-------------|
| PostDesignerAction Action (HttpPost) method for posting the request for designer actions. | |
| UploadReportAction Action (HttpPost) method for posted file actions. | |
| SetData Write the resource into storage location. | |
| GetData Read the resource from storage location. | |
| PostReportAction Action (HttpPost) method for posting the request for report process. | |
| GetResource Action (HttpGet) method for getting resource for report. | |
| OnInitReportOptions Report initialization method that is triggered when the report begins to be processed. | |
| OnReportLoaded Report loaded method that occurs when the report and sub report start loading. | |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| Methods | Description |
|------------------------------------------------------------------|-------------|
| GetResource Returns the report resource for the requested key. | |

| ProcessReport | Processes the report request and returns the result. |

ReportHelper

The class `ReportHelper` contains helper methods that help process Post or Get request for report preview action and returns the response to the Report Designer. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

Open the `ReportingAPIController` and add the following using statement.

```
'csharp
using BoldReports.Web;
using BoldReports.Web.ReportDesigner;
using BoldReports.Web.ReportViewer;
using System.IO;
using System.Reflection;
using System.Web;
'
```

Next, add the `[EnableCors]` attribute to the `ReportingAPIController` class.

```
'csharp
[System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
public class ReportingAPIController : ApiController
{
}
'
```

Inherit the `IReportDesignerController` interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
[System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
public class ReportingAPIController : ApiController, IReportDesignerController
{
    private string GetFilePath(string itemName, string key)
```

```
{  
    string targetFolder = HttpContext.Current.Server.MapPath("~/");  
    targetFolder += "Cache";  
    if (!Directory.Exists(targetFolder))  
    {  
        Directory.CreateDirectory(targetFolder);  
    }  
    if (!Directory.Exists(targetFolder + "\\\" + key))  
    {  
        Directory.CreateDirectory(targetFolder + "\\\" + key);  
    }  
    return targetFolder + "\\\" + key + "\\\" + itemName;  
}  
  
[System.Web.Http.ActionName("GetImage")]  
[AcceptVerbs("GET")]  
public object GetImage(string key, string image)  
{  
    return ReportDesignerHelper.GetImage(key, image, this);  
}  
public object PostDesignerAction(Dictionary<string, object> jsonResult)  
{  
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null);  
}  
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)  
{  
    errorMessage = string.Empty;  
    if (itemData.Data != null)  
    {  
        File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);  
    }  
    else if (itemData.PostedFile != null)  
    {  
        var fileName = itemId;
```

```
if (string.IsNullOrEmpty(itemId))
{
    fileName = Path.GetFileName(itemData.PostedFile.FileName);
}

itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));
}

return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

public void UploadReportAction()
{
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}

[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]

public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
    };
}
```

```
//Gauge component scripts
resourcesPath + @"\bold-reports\common\ej2-base.min.js",
resourcesPath + @"\bold-reports\common\ej2-data.min.js",
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
    resourcesPath + @"\jquery-1.7.1.min.js"
};

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

public object PostReportAction(Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this as IReportController);
}
}
```

Compile and run the Web API service application.

Create ASP.NET Core Web API Service

In this section, you will learn how to create a ASP.NET Core Web API for Report Designer using the new ASP.NET Core Web Application template.

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Designer API creation.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

![Creating a new ASP.NET Core Web Application Project](/static/assets/javascript/report-designer/report-service/core-api-images/aspnet-new-project.png)

Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/javascript/report-designer/report-service/core-api-images/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. In the Solution Explore, right-click the Dependencies, select Manage NuGet Packages and then search the package **BoldReports.Net.Core** and install to the application. The following provides detail of the packages and its usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core package**.

Newtonsoft.Json | Serialize and deserialize the data or report for report designer. It is a mandatory package for the report designer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer and designer. It should be referred in project when process the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0 .

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Designer NuGet packages.

Register CORS

Browser security prevents Report Designer from making requests to your API Service when both runs in a different domain. To allow access to your API service from a different domain, you must enable cross-origin requests.

Register `AddCors` in `Startup.ConfigureServices` to add CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
    {
        builder.AllowAnyOrigin()
            .AllowAnyMethod()
            .AllowAnyHeader();
    }));
    services.AddMvc();
}
```

For more information about CORS, see [Configure CORS for API](#)

Add API Service

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as `ReportingAPIController`, and then click **Add**.

![Adding a new controller to the project](/static/assets/javascript/report-designer/report-service/core-api-images/add-core-api-controller.png)

While adding API Controller class, name it with the suffix `Controller` that is mandatory.

Configure Web API

The `IReportDesignerController` interface contains the required actions and helper methods declaration to process the designer file and data actions. The `ReportDesignerHelper` and `ReportHelper` class contains methods that help to process Post or Get request from the control and return the response.

| Methods | Description | |
|---------------------------------------------------------------------------------------------|-------------|--|
| | | |
| PostDesignerAction Action (HttpPost) method for posting the request for designer actions. | | |

| | |
|----------------------------------------------------------------------------------------------------------|--|
| UploadReportAction Action (HttpPost) method for posted file actions. | |
| SetData Write the resource into storage location. | |
| GetData Read the resource from storage location. | |
| GetImage Action (HttpGet) method for getting resource for report images. | |
| PostReportAction Action (HttpPost) method for posting the request for report process. | |
| GetResource Action (HttpGet) method for getting resource for report. | |
| OnInitReportOptions Report initialization method that occurs when the report is about to be processed. | |
| OnReportLoaded Report loaded method that occurs when the report and sub report start loading. | |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| | |
|----------------------------------------------------------------------|--|
| Methods Description | |
| ----- ----- | |
| GetResource Returns the report resource for the requested key. | |
| ProcessReport Processes the report request and returns the result. | |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request for report preview action and returns the response to the Report Designer. It has the following methods.

| | |
|----------------------------------------------------------------------|--|
| Methods Description | |
| ----- ----- | |
| GetResource Returns the report resource for the requested key. | |
| ProcessReport Processes the report request and returns the result. | |

Open the **ReportingAPIController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
'
```

Add the **{action}** placeholder in the route template, if not contains with default controller attribute.

```
'csharp
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller
```

```
{  
}  
`
```

Next, add the [EnableCors] attribute to the ReportingAPIController class and specify the policy name which given in Startup.ConfigureServices.

```
`csharp  
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]  
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller,
BoldReports.Web.ReportDesigner.IReportDesignerController
{
}  
`
```

Inherit the IReportDesignerController interface, and implement its methods (replace the following code in newly created Web API controller).

```
`csharp  
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller,
BoldReports.Web.ReportDesigner.IReportDesignerController
{
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    public ReportingAPIController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }
    private string GetFilePath(string itemName, string key)
    {
        string targetFolder = this._hostingEnvironment.WebRootPath + "\\";
        targetFolder += "Cache";
```

```
if (!System.IO.Directory.Exists(targetFolder))
{
    System.IO.Directory.CreateDirectory(targetFolder);
}

if (!System.IO.Directory.Exists(targetFolder + "\\\" + key))
{
    System.IO.Directory.CreateDirectory(targetFolder + "\\\" + key);
}

return targetFolder + "\\\" + key + "\\\" + itemName;
}

public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}

public object GetResource(ReportResource resource)
{
    return ReportHelper.GetResource(resource, this, _cache);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

[HttpPost]
public object PostDesignerAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null, this._cache);
}

public object PostFormDesignerAction()
{
    return ReportDesignerHelper.ProcessDesigner(null, this, null, this._cache);
}
```

```
public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, this._cache);
}

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        System.IO.File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = System.IO.Path.GetFileName(itemData.PostedFile.FileName);
        }
        using (System.IO.MemoryStream stream = new System.IO.MemoryStream())
        {
            itemData.PostedFile.OpenReadStream().CopyTo(stream);
            byte[] bytes = stream.ToArray();
            var writePath = this.GetFilePath(fileName, key);
            System.IO.File.WriteAllBytes(writePath, bytes);
            stream.Close();
            stream.Dispose();
        }
    }
}
```

```
return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = System.IO.File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

[HttpPost]
public void UploadReportAction()
{
    ReportDesignerHelper.ProcessDesigner(null, this, this.Request.Form.Files[0], this._cache);
}
```

Compile and run the Web API service application.

How to queries for Bold Reports Report Designer

This section helps to get the answer for the frequently asked how to queries in Bold Reports Report Designer.

[Create a RDL report](#)

[Create a RDLC report](#)

[Open server report using report path on opening Report Designer](#)

[How to add datasource and dataset for Report Designer from application?](#)

Create a RDLC report

Report Definition Language Client-side, is the extension of the report file. The RDLC files does not require the SQL Server Reporting Services to process the reports.

This section describes step by step procedure to create an RDLC report in JavaScript Report Designer.

Create an application

Create a Web Report Designer javascript application, by following the steps provided in [Add Web Report Designer to a javascript application](#) section.

Set report type property

On report designer control initialization, set the `reportType` property as `RDLC`.

Add the following code in the <body> tag of the web Report Designer HTML page.

```
'html
<!-- Creating a div tag which will act as a container for ejReportDesigner widget.-->
<div style="height: 600px; width: 950px; min-height: 400px;" id="designer"></div>
<script type="text/javascript">
$(function () {
    $("#designer").ejReportDesigner({
        serviceUrl: "/api/ReportingAPI",
        reportType: ej.ReportDesigner.ReportType.RDLC
    });
});
</script>
'
```

When you run the application the report designer will be rendered like below.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/report-designer-initial-view.png)

Create data

Click the **Data** icon in the configuration panel to open a **Data** configuration panel.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/data-configuration-icon.png)

Click on the **NEW DATA** button in **DATA** panel.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/new-data-button.png)

Now, the **Data Fields** dialog will be opened like below.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/data-fields-dialog.png)

Edit the name of the **Data** in the **Name** field, if required.

To create fields for the dataset, click on the **ADD** button.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/add-field.png)

Provide the **Field Name** in the first drop-down list.

In the second drop-down list, choose the datatype for the respective field.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/create-new-field.png)

Similarly, you can create multiple fields for the dataset.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/create-multiple-fields.png)

Click on the **OK** button. Now, the dataset will be listed under the **DATA** pane like below.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/data-list-view.png)

Assign data

Drag and drop a chart report item to the design area.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/drag-and-drop-chart.png)

Switch to the **DATA** tab in the properties panel.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/data-assign-tab.png)

Select and drag the numeric column (measure element) or the numeric expression column from the **Measure** section and drop it in the **Y Values** section.

Select and drag the dimension element from the **Dimensions** section to measure against any of the selected numeric column(s) in **Y Value(s)** section, and drop into the **Column(s)** section.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/assign-fields-to-chart.png)

Now, the final design will look like below.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/chart-final-design.png)

Preview report

Click on the **Preview** button in the toolbar. Now, the following view will be displayed.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/click-on-preview-report.png)

To bind data to the widgets, provide valid JSON array collection as shown in following sample code.

```
'js
[{
  "ProdCat": "Accessories", "SubCat": "Helmets", "OrderYear": "2002", "OrderQtr": "Q1", "Sales": 4945.6925
},
{
  "ProdCat": "Components", "SubCat": "Road Frames", "OrderYear": "2002", "OrderQtr": "Q3", "Sales": 957715.1942
},
{
  "ProdCat": "Components", "SubCat": "Forks", "OrderYear": "2002", "OrderQtr": "Q4", "Sales": 23543.1060
},
{
  "ProdCat": "Bikes", "SubCat": "Road Bikes", "OrderYear": "2002", "OrderQtr": "Q1", "Sales": 3171787.6112
},
{
  "ProdCat": "Accessories", "SubCat": "Helmets", "OrderYear": "2002", "OrderQtr": "Q3", "Sales": 33853.1033
},
{
  "ProdCat": "Components", "SubCat": "Wheels", "OrderYear": "2002", "OrderQtr": "Q4", "Sales": 163921.8870
},
{
  "ProdCat": "Bikes", "SubCat": "Road Bikes", "OrderYear": "2003", "OrderQtr": "Q2", "Sales": 4119658.6506
},
{
  "ProdCat": "Clothing", "SubCat": "Socks", "OrderYear": "2003", "OrderQtr": "Q3", "Sales": 6968.6884
},
```

```
{
  "ProdCat": "Bikes", "SubCat": "Road Bikes", "OrderYear": "2003", "OrderQtr": "Q4", "Sales": 3734891.6389
},
{
  "ProdCat": "Components", "SubCat": "Mountain Frames", "OrderYear": "2002", "OrderQtr": "Q3", "Sales": 608352.8754
}
]
`
```

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/bind-json-data.png)

Now, the report preview will be displayed like below.

![Create new reporting application project](/static/assets/javascript/report-designer/images/how-to/create-rdlc-report/report-preview-final-state.png)

See Also

[Is it possible to create RDLC reports from business object datasource?](#)

How to open server report using report path at the time of opening the Report Designer

Use the `openReport` method in `index.html` file as shown in below code snippet to open a report from report server.

```
'javascript
<div style="height: 720px; width: 1500px; min-height: 400px;" id="designer"></div>
<script>
$(function () {
  $(document.body).bind('submit', $.proxy(formSubmit, this));
  $(window).bind('beforeunload', $.proxy(windowUnload, this));
  var dataValue = "";
  var apiRequest = new Object();
  apiRequest.password = "demo";
  apiRequest.userid = "guest@boldreports.com";
  $.ajax({
    type: "POST",
    url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
    ...});`
```

```
data: apiRequest,  
success: function (data) {  
    dataValue = data.Token;  
    var token = JSON.parse(dataValue);  
    $("#designer").boldReportDesigner(  
    {  
        serviceUrl: "https://on-premise-demo.boldreports.com/reporting/reportservice/api/Designer",  
        serviceAuthorizationToken: token["tokentype"] + " " + token["accesstoken"],  
        ajaxBeforeLoad: "onAjaxRequest"  
    });  
    var designerObj = $('#designer').data('boldReportDesigner');  
    designerObj.openReport('/Sample Reports/Sales Report');  
}  
});  
});  
});  
</script>  
'
```

How to add the data source and dataset for Report Designer from application

You can add the data for reports in the Report Designer from application level on initializing the Report Designer. This can be achieved using the `addDataSource` and `addDataSet` functions, by which you can add the data source and dataset for the reports at the time of initialization of Report Designer.

Find the following steps to add the data source and dataset for Report Designer from application.

Create a function and bind it with the `create` API as shown in the following code snippet.

```
`javascript  
$(function () {  
    $("#designer").boldReportDesigner({  
        serviceUrl: "/api/ReportingAPI",  
        create: "controlInitialized"  
    });  
});  
function controlInitialized(){  
...  
}
```

```
}
```

```
'
```

Use the `addDatasource` method to add the data source in Report Designer as shown in the following code snippet,

```
'html
<script>
var datasource =
{
    type:'BoldReports.RDL.DOM.DataSource',
    Name:'DataSource1',
    Transaction:false,
    DataSourceReference:null,
    SecurityType:'DataBase',
    ConnectionProperties:{
        type:'BoldReports.RDL.DOM.ConnectionProperties',
        ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
        EmbedCredentials:false,
        DataProvider:'SQL',
        IsDesignState:false,
        IntegratedSecurity:false,
        UserName:'',
        PassWord:'',
        Prompt:'Specify the Username and password for DataSource DataSource1',
        CustomProperties: []
    }
};

function controlInitialized(){
    var designerObj = $("#designer").data("boldReportDesigner");
    designerObj.addDataSource(datasource);
}
</script>
'
```

Use the `addDataSet` method to add dataset in Report Designer as shown in the following code snippet,

```
'html
<script>
var dataset =
{
  type:'BoldReports.RDL.DOM.DataSet',
  Name:'DataSet1',
  Fields:[
    { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},
    { type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},
    { type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },
    { type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null}
  ],
  Query: {
    type: "BoldReports.RDL.DOM.Query",
    CommandText: "SELECT
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[HumanResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM
[HumanResources].[Department]",
    CommandType: 0,
    DataSourceName: "DataSource1",
    QueryDesignerState: {
      type: "BoldReports.RDL.DOM.QueryDesignerState",
      Expressions: null,
      Filters: null,
      Joins: null,
      StoredProcedure: null,
      Tables: [
        {
          type: "BoldReports.RDL.DOM.Table",

```

```
Columns: [
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
  IsSelected: true, Name: "DepartmentID"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
  IsSelected: true, Name: "Name"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
  IsSelected: true, Name: "GroupName"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
  IsSelected: true, Name: "ModifiedDate"
}
],
Name: "Department",
Schema: "HumanResources",
SchemaLevels: [
{ Name: "HumanResources", SchemaType: "Schema"},

{ Name: "Tables", SchemaType: "Category"},

{ Name: "Department", SchemaType: "Table"}
]
}
]
},
QueryParameters: [],
Timeout: 0
},
CaseSensitivity:0,
Collation:null,
AccentSensitivity:0,
KanatypeSensitivity:0,
WidthSensitivity:0,
Filters:[]
```

```

SharedDataSet:null,
InterpretSubtotalsAsDetails:0,
DataSetInfo:null,
DataSetObject:null
};

function controlInitialized(){
var designerObj = $("#designer").data("boldReportDesigner");
designerObj.addDataSet(dataset);
}

</script>
`
```

Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

Breaking Changes

When you upgrade from previous versions of Bold Reports to 2.2.23, there are few breaking changes. The following section explains the breaking changes for Bold Reports version 2.2.23.

Designer resource read and write API

The resource write and read API's in `IReportDesignerController` are modified to simplify the resource write and read actions with Bold Report Designer.

Below are the new API details tabulated against old API's.

| Old API | New API | Description |
|-------------|---------|-----------------------------------------------------|
| UploadFile | SetData | Writes the designer resource into storage location. |
| GetFilePath | GetData | Reads the designer resource from storage location. |
| GetFiles | | |
| GetFile | | |

Parameters

SetData

| Parameter Name | Type | Description |
|----------------|--------|------------------------------------|
| key | string | Bold report designer unique token. |

| | | | |
|--------------|---------------|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| itemId | string | | Unique id of designer resource. |
| itemData | object | | Gets the resource data. |
| | Property Name | Type | |
| | Data | byte array | Gets the resource data as byte array. To write an external resource into storage location, pass the resource data in this property. |
| | PostedFile | HttpPostedFile | Gets the uploaded resource data as HttpPostedFile. The resource data uploaded within report designer will be received in this property. |
| errorMessage | string | | Returns the error message, if the write action is failed. |

GetData

| Parameter Name | Types | Description |
|----------------|--------|------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |

Return Type

| API Name | Return Type | | |
|----------|---------------|------------|----------------------------------------------------------|
| SetData | boolean | | |
| GetData | object | | |
| | Property Name | Type | Description |
| | Data | byte array | Contains the requested resource data as byte array. |
| | errorMessage | string | Returns the error message, if the read action is failed. |

Code snippet

| Old snippet | New snippet |
|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>UploadFilecsharppublic bool UploadFile(System.Web.HttpPostedFile httpPostedFile){ //Implement resource write actions}</pre> | <pre>SetDatacsharppublic bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage){ //Implement resource write actions}</pre> |
| <pre>GetFilescsharppublic List GetFiles(FileType fileType){ //Implement resource read actions}</pre> | |
| <pre>GetFilePathcsharppublic string GetFilePath(string fileName){ //Implement actions to get file path}</pre> | <pre>GetDatacsharppublic ResourceInfo GetData(string key, string itemId){ //Implement resource read actions}</pre> |
| <pre>GetFilecsharppublic FileModel GetFile(string filename, bool isOverride){ //Implement resource read actions}</pre> | |

JavaScript Report Designer API reference

Using the Report Designer public properties, events, and members you can change the report options programmatically. The API gives you simple access to the functionality behind the Report Designer. You can use this to create your own custom applications or to script interactions with Report Designer. The following documentation contains the Report Designer public API's details with code snippets.

[Events](#)[Members](#)[Methods](#)[Properties](#)

Members

Properties

```
<h3 class="doc-prop-wrapper" id="configurepanesettings" data-Path="configurepanesettings-
configurePaneSettings">
<a href="#configurepanesettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4
```

| | |
|---------|------------|
| Members | Properties |
|---------|------------|

3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 2.72-2 3.25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z">

</path>

</svg>

<a>configurePaneSettings

 [ConfigurePaneSettings](#)

</h3>

Shows or hides the items of configuration pane in ReportDesigner control.

<h3 class="doc-prop-wrapper" id="locale" data-Path="locale-locale">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>

</svg>

<a>locale

 string

</h3>

Specifies the locale for report designer.

Defaults to *en-US*

Example

`html

<div id="container"></div>

<script>

\$("#container").boldReportDesigner({

locale: "fr-FR"

});

</script>

`

<h3 class="doc-prop-wrapper" id="permissionsettings" data-Path="permissionsettings-permissionSettings">

```

<a href="#permissionsettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4
3h4c1.45 0 3 1.69 3 3.5 0 1.41-91 2.72-2 3.25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2
1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-
2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z">
</path>
</svg>
</a><span class='doc-prop-name'>permissionSettings</span>
<span class="doc-prop-type"> PermissionSettings
</span>
</h3>

```

Shows or hides the create, edit, and delete options in data source and dataset panels.

```

<h3 class="doc-prop-wrapper" id="previewoptions" data-Path="previewoptions-previewOptions">
<a href="#previewOptions" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4
3h4c1.45 0 3 1.69 3 3.5 0 1.41-91 2.72-2 3.25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2
1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-
2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z">
</path>
</svg>
</a><span class='doc-prop-name'>previewOptions</span>
<span class="doc-prop-type"> PreviewOptions
</span>
</h3>

```

Gets or sets the properties, events and methods of Report Viewer component for report preview action from Report Designer.

```

<h3 class="doc-prop-wrapper" id="reportdataextensions" data-Path="reportdataextensions-reportDataExtensions">
<a href="#reportdataextensions" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4
3h4c1.45 0 3 1.69 3 3.5 0 1.41-91 2.72-2 3.25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2
1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-
2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z">
</path>
</svg>
</a><span class='doc-prop-name'>reportDataExtensions</span>
<span class="doc-prop-type"> ReportDataExtensions
</span>
</h3>

```

```
</path>
</svg>
</a><span class='doc-prop-name'>reportDataExtensions</span>
<span class="doc-prop-type"> ReportDataExtensions
</span>
</h3>
```

Gets or sets the list of custom data extension items.

Defaults to []

```
<h3 class="doc-prop-wrapper" id="reportitemextensions" data-Path="reportitemextensions-reportItemExtensions">
<a href="#reportitemextensions" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4
3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 2.72-2 3.25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2
1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z">
</path>
</svg>
```

```
</a><span class='doc-prop-name'>reportItemExtensions</span>
<span class="doc-prop-type"> ReportItemExtensions
</span>
</h3>
```

Gets or sets the list of custom report items.

Defaults to []

```
<h3 class="doc-prop-wrapper" id="reportpath" data-Path="reportpath-reportPath">
<a href="#reportpath" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2
3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
</a><span class='doc-prop-name'>reportPath</span>
<span class="doc-prop-type"> string
</span>
```

</h3>

Gets or sets the report path of server.

Defaults to *null*

Example

'js

<div id="container"></div>

<script>

\$("#container").boldReportDesigner({

reportPath: "/Sample Reports/invoice"

});

</script>

'

<h3 class="doc-prop-wrapper" id="reporttype" data-Path="reporttype-reportType">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>

</svg>

reportType

 enum

</h3>

<ts name = "ej.ReportDesigner.ReportType" style = "display:none"></ts>

Gets or sets the report type.

| Name | Description |
|------|--------------------------------|
| RDL | Renders designer in RDL mode. |
| RDLC | Renders designer in RDLC mode. |

Defaults to *ej.ReportDesigner.ReportType.RDL*

Example

'js

<div id="container"></div>

```
<script>
$("#container").boldReportDesigner({
reportType: ej.ReportDesigner.ReportType.RDLC
});
</script>
` 

<h3 class="doc-prop-wrapper" id="reportserverurl" data-Path="reportserverurl-reportServerUrl">
<a href="#reportserverurl" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>reportServerUrl</span>
<span class="doc-prop-type"> string
</span>
</h3>
Gets or sets the reports server URL.
Defaults to null
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
reportServerUrl: "https://reportserver.syncfusion.com/"
});
</script>
` 

<h3 class="doc-prop-wrapper" id="serviceauthorizationtoken" data-Path="serviceauthorizationtoken-serviceAuthorizationToken">
<a href="#serviceauthorizationtoken" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

| Members | Properties |
|---------|------------|
|---------|------------|

1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>

```
</svg>
</a><span class='doc-prop-name'>serviceAuthorizationToken</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the service authorization token to access the Report Server API services.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
serviceAuthorizationToken: token['tokentype'] + '' + token['accesstoken']
});
</script>
`
```

<h3 class="doc-prop-wrapper" id="serviceurl" data-Path="serviceurl-serviceUrl">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>

```
</svg>
</a><span class='doc-prop-name'>serviceUrl</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the URL of the WebAPI service; it will be used for processing the report.

Defaults to *null*

```
<span style="font-weight:bold">Example</span>
`js
```

```

<div id="container"></div>
<script>
$("#container").boldReportDesigner({
serviceUrl: '../..../api/ReportingService'
});
</script>
` 

<h3 class="doc-prop-wrapper" id="tenantname" data-Path="tenantname-tenantName">
<a href="#tenantname" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>tenantName</span>
<span class="doc-prop-type"> string
</span>
</h3>
Gets or sets the tenant name of the user groups; it will be used when integrating report designer with report server.
Defaults to null
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
tenantName: "site"
});
</script>
` 

<h3 class="doc-prop-wrapper" id="toolbarsettings" data-Path="toolbarsettings-toolbarSettings">
<a href="#toolbarsettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4
3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 2.72-2 3.25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2
1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-
2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z">
</path>
```

```
</svg>
</a><span class='doc-prop-name'>toolbarSettings</span>
<span class="doc-prop-type"> ToolbarSettings
</span>
</h3>
```

Defines the settings of the ReportDesigner toolbar.

```
<h3 class="doc-prop-wrapper" id="waitingpopuptemplate" data-Path="waitingpopuptemplate-
waitingPopupTemplate">
<a href="#waitingpopuptemplate" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2
3.25C6 11.31 7.55 13 9 13h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>waitingPopupTemplate</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the waiting popup template for the Report designer.

Defaults to *null*

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
waitingPopupTemplate: '<div id="customLoadingtemplate" style="background-
image:url(../../load_light.gif); width: 200px; height: 15px"><p>Loading report... </p></div>'
});
</script>
```

Methods

[addDataSet](#)

Add a dataset to the report at runtime.

| Parameter | Type | Description |
|-------------------------------------------------------------------------|------|-------------|
| dataset object a JSON to define a connection properties for dataset | | |

Example

Add embedded dataset to the report

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var dataset =
{
  type:'BoldReports.RDL.DOM.DataSet',
  Name:'DataSet1',
  Fields:[
    { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},
    { type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},
    { type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },
    { type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null}
  ],
  Query: {
    type: "BoldReports.RDL.DOM.Query",
    CommandText: "SELECT
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[HumanResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate]"
  }
};'
```

```
nResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM  
[HumanResources].[Department]",  
CommandType: 0,  
DataSourceName: "DataSource1",  
QueryDesignerState: {  
    type: "BoldReports.RDL.DOM.QueryDesignerState",  
    Expressions: null,  
    Filters: null,  
    Joins: null,  
    StoredProcedure: null,  
    Tables: [  
        {  
            type: "BoldReports.RDL.DOM.Table",  
            Columns: [  
                { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
                  IsSelected: true, Name: "DepartmentID"  
                },  
                { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
                  IsSelected: true, Name: "Name"  
                },  
                { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
                  IsSelected: true, Name: "GroupName"  
                },  
                { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
                  IsSelected: true, Name: "ModifiedDate"  
                }  
            ],  
            Name: "Department",  
            Schema: "HumanResources",  
            SchemaLevels: [  
                { Name: "HumanResources", SchemaType: "Schema"},  
                { Name: "Tables", SchemaType: "Category"},  
                { Name: "Department", SchemaType: "Table"}  
            ]  
        }  
    ]  
}
```

| | |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <p>Methods</p> <pre>] }] }, QueryParameters: [], Timeout: 0 }, CaseSensitivity:0, Collation:null, AccentSensitivity:0, KanatypeSensitivity:0, WidthSensitivity:0, Filters:[], SharedDataSet:null, InterpretSubtotalsAsDetails:0, DataSetInfo:null, DataSetObject:null }; designerObj.addDataSet(dataset); </script> `</pre> | <p>addDataSet</p> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|

Add shared dataset to the report

```

`html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var dataset =
{
    type:'BoldReports.RDL.DOM.DataSet',
    Name:'DataSet1',
```

Methods addDataSource

Fields:[

```
{ type: "BoldReports.RDL.DOM.Field", DataField: "ProdCat", Name: "ProdCat", TypeName: "System.String", Value: null},  
{ type: "BoldReports.RDL.DOM.Field", DataField: "SubCat", Name: "SubCat", TypeName: "System.String", Value: null},  
{ type: "BoldReports.RDL.DOM.Field", DataField: "OrderYear", Name: "OrderYear", TypeName: "System.Int32", Value: null },  
{ type: "BoldReports.RDL.DOM.Field", DataField: "OrderQtr", Name: "OrderQtr", TypeName: "System.String", Value: null},  
{ type: "BoldReports.RDL.DOM.Field", DataField: "Sales", Name: "Sales", TypeName: "System.Decimal", Value: null}  
],  
Query:null,  
CaseSensitivity:0,  
Collation:null,  
AccentSensitivity:0,  
KanatypeSensitivity:0,  
WidthSensitivity:0,  
Filters:[],  
SharedDataSet: {  
    type: "BoldReports.RDL.DOM.SharedDataSet",  
    QueryParameters: [],  
    SharedDataSetReference: 'Sales'  
},  
InterpretSubtotalsAsDetails:0,  
DataSetInfo:null,  
DataSetObject:null  
};  
designerObj.addDataSet(dataset);  
</script>  
,
```

[addDataSource](#)

Add a datasource to the report at runtime.

| Parameter | Type | Description | |
|-----------|------|-------------|--|
| | | | |

| datasource | object | a JSON to define a connection properties for datasource |
Example

Add embedded datasource to the report

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var datasource =
{
    type:'BoldReports.RDL.DOM.DataSource',
    Name:'DataSource1',
    Transaction:false,
    DataSourceReference:null,
    SecurityType:'DataBase',
    ConnectionProperties:{
        type:'BoldReports.RDL.DOM.ConnectionProperties',
        ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
        EmbedCredentials:false,
        DataProvider:'SQL',
        IsDesignState:false,
        IntegratedSecurity:false,
        UserName:'',
        PassWord:'',
        Prompt:'Specify the Username and password for DataSource DataSource1',
        CustomProperties: []
    }
};
designerObj.addDataSource(datasource);
</script>
`
```

Methods

addItem

Add shared datasource to the report

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var datasource =
{
    type:'BoldReports.RDL.DOM.DataSource',
    Name:'DataSource1',
    Transaction:false,
    DataSourceReference: 'AdventureWorks',
    SecurityType:'None',
    ConnectionProperties:null
};
designerObj.addDataSource(datasource);
</script>
'
```

addItem

Add a report item to the report at runtime.

| Name | Type | Description | | |
|------|--------|-------------------------------------------|--------|-------------------------------------------------------------------|
| item | object | JSON for the new report item to be added. | | |
| | | Name | Type | Description |
| | | left | number | Set the left position value to the report item. |
| | | top | number | Set the top position value to the report item. |
| | | itemType | string | Set the type of the report item. Input Text TextBox |

| Name | Type | Description | |
|------|------------|-------------|--------------------------------------------------------|
| | | Image | |
| | | Line | |
| | | Rectangle | |
| | | Chart | |
| | | Table | |
| | | List | |
| | | Grid | |
| | | PivotGrid | |
| | | SubReport | |
| | | Custom | |
| | designArea | string | Set the target to render report item. |
| | parentName | string | Set the name of the parent item. |
| | rowIndex | number | Set the index of the row in table/list report item. |
| | colIndex | number | Set the index of the column in table/list report item. |

Examples

Add a report item to report body

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var item = {
left: 300, top: 100, itemType: 'Tablix',
designArea: ej.ReportDesigner.DesignArea.Body,
```

```
parentName : null,  
rowIndex: -1, colIndex: -1  
};  
designerObj.addItem(item);  
</script>  
'
```

Add a report item to report header

```
`html  
<div id="container"></div>  
<script>  
//Create Report Designer Instance  
$("#container").boldReportDesigner();  
var designerObj = $("#container").data("boldReportDesigner");  
var item = {  
left: 100, top: 30, itemType: 'Image',  
designArea: ej.ReportDesigner.DesignArea.Header,  
parentName : null,  
rowIndex: -1, colIndex: -1  
};  
designerObj.addItem(item);  
</script>  
'
```

Add a report item to report footer

```
`html  
<div id="container"></div>  
<script>  
//Create Report Designer Instance  
$("#container").boldReportDesigner();  
var designerObj = $("#container").data("boldReportDesigner");  
var item = {  
left: 50, top: 50, itemType: 'TextBox',  
designArea: ej.ReportDesigner.DesignArea.Footer,
```

```
parentName : null,  
rowIndex: -1, colIndex: -1  
};  
designerObj.addItem(item);  
</script>  
'
```

Add a report item into a tablix cell

```
'html  
<div id="container"></div>  
<script>  
//Create Report Designer Instance  
$("#container").boldReportDesigner();  
var designerObj = $("#container").data("boldReportDesigner");  
//Add tablix item  
var item = {  
    left: 50, top: 50, itemType: 'Tablix',  
    designArea: ej.ReportDesigner.DesignArea.Body,  
    parentName:null,  
    rowIndex: -1, colIndex: -1  
};  
designerObj.addItem(item);  
//Add image report item into tablix cell  
var item = {  
    left: 50, top: 50, itemType: 'Image',  
    designArea: ej.ReportDesigner.DesignArea.Body,  
    parentName: 'Tablix1',  
    rowIndex: 1, colIndex: 0  
};  
designerObj.addItem(item);  
</script>  
'
```

Add a report item into a rectangle report item

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
//Add rectangle item
var item = {
left: 50, top: 50, itemType: 'Rectangle',
designArea: ej.ReportDesigner.DesignArea.Body,
parentName:null,
rowIndex: -1, colIndex: -1
};
designerObj.addItem(item);
//Add image report item into rectangle item
var item = {
left: 10, top: 20, itemType: 'Image',
designArea: ej.ReportDesigner.DesignArea.Body,
parentName: 'Rectangle1',
rowIndex: -1, colIndex: -1
};
designerObj.addItem(item);
</script>
`
```

Add a chart report item to report at runtime

By default, **Column** chart type will be added to the report.

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
```

```
var item = {  
    left: 50, top: 50, itemType: 'Chart',  
    designArea: ej.ReportDesigner.DesignArea.Body,  
    parentName : null,  
    rowIndex: -1, colIndex: -1  
};  
designerObj.addItem(item);  
</script>  
'
```

Add a **Pie** chart report item to report at runtime

To add specific chart type to the report, follow the syntax : **Chart-[chart-type]**. Refer [Chart Types](#) sections for details about supported chart types.

```
'html  
<div id="container"></div>  
<script>  
//Create Report Designer Instance  
$("#container").boldReportDesigner();  
var designerObj = $("#container").data("boldReportDesigner");  
var item = {  
    left: 50, top: 50, itemType: 'Chart-Pie',  
    designArea: ej.ReportDesigner.DesignArea.Body,  
    parentName : null,  
    rowIndex: -1, colIndex: -1  
};  
designerObj.addItem(item);  
</script>  
'
```

Add a custom report item to report at runtime

To add specific custom report item type to the report, follow the syntax : **Custom-[custom-report-item-name]**. Refer [reportItemExtensions](#) property to render custom report item in Report Designer.

```
'html  
<div id="container"></div>
```

Methods bringForward

```
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var item = {
    left: 50, top: 50, itemType: 'Custom-Barcode',
    designArea: ej.ReportDesigner.DesignArea.Body,
    parentName : null,
    rowIndex: -1, colIndex: -1
};
designerObj.addItem(item);
</script>
`
```

bringForward

Visually move the selected report item over its closest intersected report items.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.bringForward();
</script>
`
```

bringToFront

Visually move the selected report item over all other intersected report items.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
```

| | |
|---------|---------|
| Methods | canCopy |
|---------|---------|

```
designerObj.bringToFront();
</script>
`
```

canCopy

Determines whether a copy operation is possible.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.canCopy();
</script>
`
```

canCut

Determines whether a cut operation is possible.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.canCut();
</script>
`
```

canPaste

Determines whether a paste operation is possible.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
```

| | |
|---------|---------|
| Methods | canRedo |
|---------|---------|

```
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.canPaste();
</script>
`
```

canRedo

Returns the boolean value indicating whether the user can redo the previous action in the report.

Returns *boolean*

Example

```
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.canRedo();
</script>
`
```

canRemove

Determines whether a delete operation is possible.

Returns *boolean*

Example

```
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.canRemove();
</script>
`
```

canUndo

Returns a boolean value indicating whether the user can undo the previous action in the report.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.canUndo();
</script>
`
```

cloneDataSet

Clone the existing dataset in the report at runtime.

| Parameter | Type | Description |
|-----------|------|-------------|
|-----------|------|-------------|

| | | |
|-------|-------|-------|
| ----- | ----- | ----- |
|-------|-------|-------|

| |
|--------------------------------------------------------|
| name string Name of the existing dataset in report |
|--------------------------------------------------------|

```
<span style="font-weight:bold">Example</span>
```

`html

```
<div id="container"></div>
```

```
<script>
```

```
//Create Report Designer Instance
```

```
$("#container").boldReportDesigner();
```

```
var designerObj = $("#container").data("boldReportDesigner");
```

```
designerObj.cloneDataSet('DataSet1');
```

```
</script>
```

`

cloneDataSource

Clone the existing datasource in the report at runtime.

| Parameter | Type | Description |
|-----------|------|-------------|
|-----------|------|-------------|

| | | |
|-------|-------|-------|
| ----- | ----- | ----- |
|-------|-------|-------|

| |
|-----------------------------------------------------------|
| name string Name of the existing datasource in report |
|-----------------------------------------------------------|

```
<span style="font-weight:bold">Example</span>
```

Methods copy

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.cloneDataSource('DataSource1');
</script>
`
```

[copy](#)

Copies the selected report item from design panel to the Report Designer internal clipboard.

Example

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.copy();
</script>
`
```

[cut](#)

Cuts the selected report item from design panel to the Report Designer internal clipboard.

Example

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.cut();
</script>
`
```

| | |
|---------|------------------|
| Methods | hasReportChanges |
|---------|------------------|

hasReportChanges

Returns the boolean value that specifies whether the report has changes or not.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.hasReportChanges();
</script>
`
```

isNewReport

Returns the boolean value that specifies whether the currently processing report is a new report or not.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.isNewReport();
</script>
`
```

isNewServerReport

Returns the boolean value that specifies whether the currently processing report is a new server report or not.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
```

```
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.isNewServerReport();
</script>
`
```

isServerReport

Returns the boolean value that specifies whether the currently processing report is obtained from the server or local.

Returns *boolean*

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.isServerReport();
</script>
`
```

newReport

To create a new report.

| Parameter | Type | Description |
|-----------------------------------------------------------------------|------|-------------|
| name (<i>optional</i>) string Name of the new report | | |
| dataSetPath (<i>optional</i>) string Name of the shared dataset | | |

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.newReport();
```

```
</script>
```

```
\
```

[newServerReport](#)

To create a new report in the report server.

| Parameter | Type | Description |
|-----------------------------------------------------------------------|------|-------------|
| | | |
| name (<i>optional</i>) string Name of the new report | | |
| dataSetPath (<i>optional</i>) string Name of the shared dataset | | |

```
<span style="font-weight:bold">Example</span>
```

```
'html
```

```
<div id="container"></div>
```

```
<script>
```

```
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.newServerReport('Test1', 'ab018ae7-f747-49a1-9e26-759e35c0a0db');
```

```
</script>
```

```
\
```

[openReport](#)

This method opens the report from the report server.

| Parameter | Type | Description |
|----------------------------------------------------------------------------|------|-------------|
| | | |
| reportPath (<i>optional</i>) string Path of the report server report | | |
| serverUrl (<i>optional</i>) string Reports server URL | | |

```
<span style="font-weight:bold">Example</span>
```

```
'html
```

```
<div id="container"></div>
```

```
<script>
```

```
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.openReport();
</script>
```

openReportDefinition

This method opens the report using raw report data.

| Parameter | Type | Description |
|-----------|----------------------|-------------------------------------------------------------------|
| rdlData | string | object | Provide the report definition in the JSON or string or XML format |

```

<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var rdlData = ""; // Assign a report data in JSON, string or XML format
designerObj.openReportDefinition(rdlData);
</script>
`
```

openReportFromDevice

Opens the client browse dialog to browse the report.

| Parameter | Type | Description |
|-----------|----------------------|-------------------------------------------------------------------|
| rdlData | string | object | Provide the report definition in the JSON or string or XML format |

```

<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.openReportFromDevice();
</script>
`
```

openServerReportDialog

Opens the report designer browse dialog to browse the available reports in the report server.

| Parameter | Type | Description |
|-----------|----------------------|-------------------------------------------------------------------|
| rdlData | string | object | Provide the report definition in the JSON or string or XML format |

```

<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
```

| | |
|---------|-------|
| Methods | paste |
|---------|-------|

```
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.openServerReportDialog();
</script>
`
```

[paste](#)

Pastes the selected report item from the Report Designer internal clipboard to design panel.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.paste();
</script>
`
```

[redo](#)

Reverses the action of the last Undo command.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.redo();
</script>
`
```

[remove](#)

Deletes the selected report item from the report.

```
<span style="font-weight:bold">Example</span>
```

Methods

removeDataSet

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.remove();
</script>
'
```

removeDataSet

Remove a dataset from the report at runtime.

| Parameter | Type | Description |
|-----------|--------|---------------------|
| dataset | string | Name of the dataset |

Example

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.removeDataSet('DataSet1');
</script>
'
```

removeDatasource

Remove a datasource from the report at runtime.

| Parameter | Type | Description |
|------------|--------|------------------------|
| datasource | string | Name of the datasource |

Example

```
'html
<div id="container"></div>
<script>
```

Methods

removeItem

```
//Create Report Designer Instance
$("#container").boldReportDesigner();

var designerObj = $("#container").data("boldReportDesigner");
designerObj.removeDatasource('DataSource1');

</script>
`
```

removeItem

Remove the given report item from the report.

| Parameter | Type | Description |
|-----------|--------|-------------------------------------------------------|
| itemName | string | Name of the report item to be removed from the report |

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();

var designerObj = $("#container").data("boldReportDesigner");
designerObj.removeItem('Tablix1');

</script>
`
```

saveReport

This method saves the report into the report server.

| Parameter | Type | Description |
|--------------------------------|--------|----------------------------------|
| reportPath (<i>optional</i>) | string | Path of the report server report |

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();

var designerObj = $("#container").data("boldReportDesigner");
```

Methods

saveReportDefinition

```
designerObj.saveReport();
</script>
`
```

saveReportDefinition

This method returns the report in JSON or XML format.

| Parameter | Type | Description | | | | | | |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|-------------|------|-----------------------------------------|-----|---------------------------------------|-------------------------------------------------------|
| callback | function | Callback method to return the report data. | | | | | | |
| type (optional) | enum <table border="1"><thead><tr><th>Name</th><th>Description</th></tr></thead><tbody><tr><td>JSON</td><td>Returns the report data in JSON format.</td></tr><tr><td>XML</td><td>Return the report data in XML format.</td></tr></tbody></table> | Name | Description | JSON | Returns the report data in JSON format. | XML | Return the report data in XML format. | Specify the format as JSON or XML to save the report. |
| Name | Description | | | | | | | |
| JSON | Returns the report data in JSON format. | | | | | | | |
| XML | Return the report data in XML format. | | | | | | | |

Example

To save the report in JSON format.

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.saveReportDefinition((args: any) => {}, ej.ReportDesigner.DataFormat.JSON);
</script>
`
```

To save the report in XML format.

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
```

Methods

saveServerReportDialog

```
var designerObj = $("#container").data("boldReportDesigner");
designerObj.saveReportDefinition((args: any) => {}, ej.ReportDesigner.DataFormat.XML);
</script>
`
```

saveServerReportDialog

Opens the report designer browse dialog to save the report into report server.

Example

```
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.saveServerReportDialog();
</script>
`
```

saveToDevice

To download the designed report.

Example

```
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.saveToDevice();
</script>
`
```

selectItems

Update the selection to report item at runtime.

| Parameter | Type | Description |
|-----------|-------|--------------------------|
| itemNames | array | Name of the report items |

Example

| | |
|---------|--------------|
| Methods | sendBackward |
|---------|--------------|

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var itemNames = ['Tablix1','Chart1','Rectangle2']
designerObj.selectItems(itemNames);
</script>
`
```

sendBackward

Visually move the selected report item behind its closest intersected report item.

Example

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.sendBackward();
</script>
`
```

sendToBack

Visually move the selected report item behind all other intersected report items.

Example

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.sendToBack();
</script>
`
```

[showDesign](#)

Performs switch action from viewer to designer at runtime.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.showDesign();
</script>
`
```

[showNewReportDialog](#)

Opens the new report dialog.

| Parameter | Type | Description |
|-----------|----------|----------------------------------------------|
| callback | function | Callback method of new report dialog actions |

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.showNewReportDialog((args: any) => {
});
</script>
`
```

[showOpenSaveReportDialog](#)

Opens the report designer browse dialog to open/save reports in the report server.

| Parameter | Type | Description |
|------------|------|-------------------------------------------------------------------------|
| browseType | enum | Mention the type as Open to perform open report action, otherwise Save. |

| Parameter | Type | | Description |
|------------|----------|---------------------------------------------------------------|----------------------------------------------|
| | Name | Description | |
| | Open | Mention the browse type as Open to launch open report dialog. | |
| | Save | Mention the browse type as Save to launch save report dialog. | |
| callback | function | | Callback method of open/save dialog actions. |
| reportName | string | | Name of the report to save. |

Example

To launch open report dialog

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var browseType = ej.ReportDesigner.BrowseType.Open;
designerObj.showOpenSaveReportDialog(browseType, (args: any) => {
});
</script>
'
```

To launch save report dialog

```
'html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
```

Methods showPreview

```
var browseType = ej.ReportDesigner.BrowseType.Save;
designerObj.showOpenSaveReportDialog(browseType, (args: any) => {
});
</script>
`
```

showPreview

Performs switch action from designer to viewer at runtime.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
// Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.showPreview();
</script>
`
```

undo

Reverses the last action that was performed.

```
<span style="font-weight:bold">Example</span>
`html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
designerObj.undo();
</script>
`
```

updateDataset

Updates the existing dataset in the report at runtime.

| Parameter | Type | Description | |
|-----------------------------------------------------|------|-------------|--|
| ----- ----- ----- | | | |
| datasetName string Name of the existing dataset | | | |

Methods updateDataset

| dataset | object | a JSON to define a connection properties for dataset |

Example

Update embedded dataset

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
var dataset = {
    type:'BoldReports.RDL.DOM.DataSet',
    Name:'DataSet2',
    Fields:[
        { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},
        { type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},
        { type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },
        { type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null}
    ],
    Query: {
        type: "BoldReports.RDL.DOM.Query",
        CommandText: "SELECT
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[HumanResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM
[HumanResources].[Department]",
        CommandType: 0,
        DataSourceName: "DataSource1",
        QueryDesignerState: {
            type: "BoldReports.RDL.DOM.QueryDesignerState",
            Expressions: null,
            Filters: null,
```

| | |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|
| <p>Methods</p> <p>Joins: null,</p> <p>StoredProcedure: null,</p> <p>Tables: [</p> <p>{</p> <p>type: "BoldReports.RDL.DOM.Table",</p> <p>Columns: [</p> <p>{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "DepartmentID" }</p> <p>{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "Name" }</p> <p>{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "GroupName" }</p> <p>{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "ModifiedDate" }</p> <p>}</p> <p>],</p> <p>Name: "Department",</p> <p>Schema: "HumanResources",</p> <p>SchemaLevels: [</p> <p>{ Name: "HumanResources", SchemaType: "Schema" },</p> <p>{ Name: "Tables", SchemaType: "Category" },</p> <p>{ Name: "Department", SchemaType: "Table" }</p> <p>]</p> <p>}</p> <p>]</p> <p>},</p> <p>QueryParameters: [],</p> <p>Timeout: 0</p> <p>},</p> <p>CaseSensitivity: 0,</p> | <p>updateDataset</p> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|

Methods updateDataset

```
Collation:null,  
AccentSensitivity:0,  
KanatypeSensitivity:0,  
WidthSensitivity:0,  
Filters:[],  
SharedDataSet:null,  
InterpretSubtotalsAsDetails:0,  
DataSetInfo:null,  
DataSetObject:null  
};  
designerObj.updateDataset('DataSet1',dataset);  
</script>  
'
```

Update shared dataset

```
`html  
<div id="container"></div>  
<script>  
//Create Report Designer Instance  
$("#container").boldReportDesigner();  
var designerObj = $("#container").data("boldReportDesigner");  
var dataset =  
{  
type:'BoldReports.RDL.DOM.DataSet',  
Name:'DataSet2',  
Fields:[  
{ type: "BoldReports.RDL.DOM.Field", DataField: "ProdCat", Name: "ProdCat", TypeName:  
"System.String", Value: null},  
{ type: "BoldReports.RDL.DOM.Field", DataField: "SubCat", Name: "SubCat", TypeName: "System.String",  
Value: null},  
{ type: "BoldReports.RDL.DOM.Field", DataField: "OrderYear", Name: "OrderYear", TypeName:  
"System.Int32", Value: null },  
{ type: "BoldReports.RDL.DOM.Field", DataField: "OrderQtr", Name: "OrderQtr", TypeName:  
"System.String", Value: null},
```

Methods updateDatasource

```
{ type: "BoldReports.RDL.DOM.Field", DataField: "Sales", Name: "Sales", TypeName: "System.Decimal",  
Value: null}  
],  
Query:null,  
CaseSensitivity:0,  
Collation:null,  
AccentSensitivity:0,  
KanatypeSensitivity:0,  
WidthSensitivity:0,  
Filters:[],  
SharedDataSet: {  
    type: "BoldReports.RDL.DOM.SharedDataSet",  
    QueryParameters: [],  
    SharedDataSetReference: 'Sales'  
},  
InterpretSubtotalsAsDetails:0,  
DataSetInfo:null,  
DataSetObject:null  
};  
designerObj.updateDataset('DataSet1',dataset);  
</script>  
'
```

updateDatasource

Updates the existing datasource in the report at runtime.

| Parameter | Type | Description |
|----------------|--------|---------------------------------------------------------|
| ----- | ----- | ----- |
| datasourceName | string | Name of the existing datasource |
| datasource | object | a JSON to define a connection properties for datasource |

Example

Update embedded datasource

```
`html  
<div id="container"></div>  
<script>
```

Methods updateDatasource

```
//Create Report Designer Instance
$("#container").boldReportDesigner();

var designerObj = $("#container").data("boldReportDesigner");
var datasource = {
    type:'BoldReports.RDL.DOM.DataSource',
    Name:'DataSource2',
    Transaction:false,
    DataSourceReference:null,
    SecurityType:'DataBase',
    ConnectionProperties:{
        type:'BoldReports.RDL.DOM.ConnectionProperties',
        ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
        EmbedCredentials:false,
        DataProvider:'SQL',
        IsDesignState:false,
        IntegratedSecurity:false,
        UserName:'',
        PassWord:'',
        Prompt:'Specify the Username and password for DataSource DataSource2',
        CustomProperties: []
    }
};

designerObj.updateDatasource('DataSource1',datasource);
</script>
`
```

Update shared datasource

```
'html
<div id="container"></div>
<script>
//Create Report Designer Instance
$("#container").boldReportDesigner();
var designerObj = $("#container").data("boldReportDesigner");
```

Events ajaxBeforeLoad

```
var datasource =  
{  
    type:'BoldReports.RDL.DOM.DataSource',  
    Name:'DataSource2',  
    Transaction:false,  
    DataSourceReference: 'AdventureWorks',  
    SecurityType:'None',  
    ConnectionProperties:null  
};  
designerObj.updateDatasource('DataSource1',datasource);  
</script>  
'
```

Events

ajaxBeforeLoad

This event will be triggered before AJAX loads.

| Arguments | Type | Description |
|---------------------------|--------|-------------------------------------------------------------------|
| headers | array | AJAX headers, we can pass any custom header through this property |
| data | object | To pass the custom data while AJAX post back |
| reportDesignerToken | string | Token of report designer |
| serviceAuthorizationToken | string | Token of ReportingService |
| actionType | string | Action type of AJAX call back |
| model | object | Returns the report designer model |

Example

```
'js  
$("#container").boldReportDesigner({  
    ajaxBeforeLoad: function(args) {  
        if (args && args.headers) {  
            args.headers.push({ 'Key': 'keyCode', 'Value': ("Authorization") });  
        }  
    }  
});  
'
```

ajaxError

This event will be triggered when AJAX result is failed.

| Arguments | Type | Description |
|---------------------------|--------|-----------------------------------|
| headers | array | Returns the AJAX headers |
| data | object | Returns the failure data |
| reportDesignerToken | string | Token of report designer |
| serviceAuthorizationToken | string | Token of ReportingService |
| actionType | string | Action type of AJAX call back |
| model | object | Returns the report designer model |

Example

```
`js
$("#container").boldReportDesigner({
  ajaxError: function(args) {
    // Write your block of code
  }
});`
```

ajaxSuccess

This event will be triggered when AJAX result is succeeded.

| Arguments | Type | Description |
|---------------------------|--------|-----------------------------------|
| headers | array | Returns the AJAX headers |
| data | object | Returns the success data |
| reportDesignerToken | string | Token of report designer |
| serviceAuthorizationToken | string | Token of ReportingService |
| actionType | string | Action type of AJAX call back |
| model | object | Returns the report designer model |

Example

```
`js
$("#container").boldReportDesigner({
  ajaxSuccess: function(args) {
    // Write your block of code
  }
});`
```

Events create

}

});

\

create

This event will be triggered when the Report Designer widget is created.

| Arguments | Type | Description |
|-----------|---------|----------------------------------------------------------|
| cancel | boolean | true, if the event should be canceled; otherwise, false. |
| model | object | Returns the report designer model |
| type | string | Returns the name of the event |

Example

`js

```
$("#container").boldReportDesigner({  
    create: function(args) {  
        // Write your block of code  
    }  
});  
\
```

destroy

This event will be triggered when the Report Designer widget is destroyed.

| Arguments | Type | Description |
|-----------|---------|----------------------------------------------------------|
| cancel | boolean | true, if the event should be canceled; otherwise, false. |
| model | object | Returns the report designer model |
| type | string | Returns the name of the event |

Example

`js

```
$("#container").boldReportDesigner({  
    destroy: function(args) {  
        // Write your block of code  
    }  
});  
\
```

newDataClick

This event will be triggered while initiating new data click action. You can suppress the new data creation panel and perform custom actions.

| Arguments | Type | Description |
|-----------|---------|----------------------------------------------------------|
| cancel | boolean | true, if the event should be canceled; otherwise, false. |
| model | object | Returns the report designer model |
| type | string | Returns the name of the event |

Example

```
'js
$("#container").boldReportDesigner({
    newDataClick: function(args) {
        // Write your block of code
    }
});'
```

openReportClick

This event will be triggered while clicking open menu items.

| Arguments | Type | Description |
|-----------|--------|---------------------------|
| target | jQuery | DOM of the clicked target |
| select | string | Name of selected item |

Example

```
'js
$("#container").boldReportDesigner({
    openReportClick: function(args) {
        // Write your block of code
    }
});'
```

previewReport

This event will be triggered while previewing the report in RDLC mode. It can be used to suppress the preview data dialog in RDLC mode.

| Arguments | Type | Description |
|-----------|------|-------------|
|-----------|------|-------------|

| Arguments | Type | Description |
|-----------------------------------------------|-------------------------------------------------------|-------------|
| reportViewer object | Contains the instance of Report Viewer component | |
| cancelDataInputDialog boolean | Specifies whether to show or hide preview data dialog | |
| dataSets array | Contains the required data to load the report | |
| Example | | |
| `js | | |
| \$("#container").boldReportDesigner({ | | |
| previewReport: function(args) { | | |
| // Write your block of code | | |
| } | | |
| }); | | |
| , | | |

reportModified

This event will be triggered when the report is modified.

| Arguments | Type | Description |
|-----------------------------------------------|-------------------------------------------------|-------------|
| isModified boolean | Specifies whether the report is modified or not | |
| reportName string | Name of Opened Report | |
| Example | | |
| `js | | |
| \$("#container").boldReportDesigner({ | | |
| reportModified: function(args) { | | |
| // Write your block of code | | |
| } | | |
| }); | | |
| , | | |

reportOpened

This event will be triggered when the report is opened.

| Arguments | Type | Description |
|-----------------------------------------------|-------------------------------------------------------|-------------|
| isServerReport boolean | Specifies whether report opened from device or server | |
| reportName string | Name of Opened Report | |
| Example | | |

Events

reportSaved

```
`js
$("#container").boldReportDesigner({
reportOpened: function(args) {
// Write your block of code
}
});
`
```

reportSaved

This event will be triggered when the report is saved.

| Arguments | Type | Description |
|----------------|---------|-------------------------------------------------------|
| isServerReport | boolean | Specifies whether report opened from device or server |
| reportAction | string | States whether report is downloaded from ReportServer |

Example

```
`js
$("#container").boldReportDesigner({
reportSaved: function (args) {
// Write your block of code
}
});
`
```

saveReportClick

This event will be triggered when the save menu items are clicked.

| Arguments | Type | Description |
|-----------|--------|---------------------------|
| target | jQuery | DOM of the clicked target |
| select | string | Name of selected item |

Example

```
`js
$("#container").boldReportDesigner({
saveReportClick: function(args) {
// Write your block of code
}
});
```

```
});
```

```
'
```

toolbarClick

This event will be triggered while clicking the toolbar items.

| Arguments | Type | Description |
|-------------------|--------|---------------------------|
| ----- ----- ----- | | |
| target | jQuery | DOM of the clicked target |
| click | string | Name of clicked item |

Example

```
`js
$("#container").boldReportDesigner({
  toolbarClick: function(args) {
    // Write your block of code
  }
});
```

toolbarRendering

This event will be triggered on rendering the Report Designer toolbar.

| Arguments | Type | Description |
|-------------------|---------|----------------------------------------------------------|
| ----- ----- ----- | | |
| cancel | boolean | true, if the event should be canceled; otherwise, false. |
| target | jQuery | Returns toolbar element. |
| model | object | Returns the report designer model |
| type | string | Returns the name of the event |

Example

```
`js
$("#container").boldReportDesigner({
  toolbarRendering: function(args) {
    // Write your block of code
  }
});`
```

JavaScript Report Designer Properties

This section provides the JavaScript Report Designer properties with example code snippets which can be used to customize the report browse, open, edit and save actions and much more.

[Configure Pane Settings](#)

[Report Data Extensions](#)

[Report Item Extensions](#)

[Toolbar Settings](#)

[Page Settings](#)

[Preview Options](#)

[Parameters](#)

[Export Settings](#)

[Preview Toolbar Settings](#)

[Parameter Settings](#)

[Data Sources](#)

[Permission Settings](#)

configurePaneSettings

Properties

```
<h3 class="doc-prop-wrapper" id="items" data-Path="items-items">
<a href="#items" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>items</span>
<span class="doc-prop-type"> enum
</span>
</h3>
```

<ts name = "ej.ReportDesigner.ConfigureItems" style = "display:none"></ts>

Shows or hides the grouped items in the configuration pane with the help of ej.ReportDesigner.ConfigureItems enum.

| Name | Description |
|--------------|--------------------------------------------------------------|
| Property | Shows or hides the properties panel in configuration pane |
| Data | Shows or hides the data panel in configuration pane |
| Parameter | Shows or hides the parameter panel in configuration pane |
| ImageManager | Shows or hides the image manager panel in configuration pane |
| All | Shows all the configuration pane items |

Defaults to *ej.ReportDesigner.ConfigureItems.All*

Example

Show all configure pane items.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
configurePaneSettings: { items: ej.ReportDesigner.ConfigureItems.All }
});
</script>
`
```

Hide all configure pane items.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
configurePaneSettings: { items: ~ej.ReportDesigner.ConfigureItems.All }
});
</script>
`
```

Hide **Properties** panel from configure pane items.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
configurePaneSettings: { items: ej.ReportDesigner.ConfigureItems.All &
~ej.ReportDesigner.ConfigureItems.Property}
});
</script>
'
```

Hide all items except **Data** panel from configure pane items.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
configurePaneSettings: { items: ej.ReportDesigner.ConfigureItems.All &
~ej.ReportDesigner.ConfigureItems.Property & ~ej.ReportDesigner.ConfigureItems.Parameter &
~ej.ReportDesigner.ConfigureItems.ImageManager}
});
</script>
'

<h3 class="doc-prop-wrapper" id="showpane" data-Path="items-showPane">
<a href="#showpane" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>showPane</span>
<span class="doc-prop-type"> boolean
</span>
</h3>
```

Shows or hides the configuration pane in ReportDesigner control.

Defaults to *true*

reportDataExtensions `array`

Properties

Example

Show configure pane.

```
`html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
configurePaneSettings: { showPane: true }
});
</script>
`
```

Hide configure pane.

```
`html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
configurePaneSettings: { showPane: false }
});
</script>
`
```

reportDataExtensions `array`

Properties

```
<h3 class="doc-prop-wrapper" id="name" data-Path="name-name">
<a href="#name" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>name</span>
<span class="doc-prop-type"> string
</span>
```

</h3>

Gets or sets the name of the datasource type.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="classname" data-Path="classname-className">
<a href="#classname" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>className</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the class name of the data extension.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="imageclass" data-Path="imageclass-imageClass">
<a href="#imageclass" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>imageClass</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the image class name to load image in data pane tile.

```
<h3 class="doc-prop-wrapper" id="displayname" data-Path="displayname-displayName">
<a href="#displayname" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>  
</a><span class='doc-prop-name'>displayName</span>  
<span class="doc-prop-type"> string  
</span>  
</h3>
```

Gets or sets the name for data extension item to display in the data pane tile.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
```

Add a custom data extension to report designer

```
`html  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
reportDataExtensions: [{  
className: 'WebAPIDataSource',  
name: 'WebAPI',  
imageClass: 'e-reportdesigner-datasource-webapi',  
displayName: 'WebAPI'  
}]  
});  
</script>  
'
```

Add multiple custom data extensions to report designer

```
`html  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
reportDataExtensions: [  
{  
className: 'WebAPIDataSource',
```

reportItemExtensions `array` Properties

```
name: 'WebAPI',
imageClass: 'e-reportdesigner-datasource-webapi',
displayName: 'WebAPI'
},
{
className: 'PSQLDataSource',
name: 'PostgreSQL',
imageClass: 'e-reportdesigner-datasource-psql',
displayName: 'PostgreSQL'
}]
});
</script>
`
```

reportItemExtensions `array`

Properties

```
<h3 class="doc-prop-wrapper" id="name" data-Path="name-name">
<a href="#name" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>name</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the name for the report item.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="classname" data-Path="classname-className">
<a href="#classname" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

</svg>
className
string

</h3>

Gets or sets the class name of the report item.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="imageclass" data-Path="imageclass-imageClass">  

<a href="#imageclass" aria-hidden="true" class="anchor">  

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-  
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

</svg>
imageClass
string

</h3>

Gets or sets the image class name to load image in widgets pane tile.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="displayname" data-Path="displayname-displayName">  

<a href="#displayname" aria-hidden="true" class="anchor">  

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-  
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

</svg>
displayName
string

</h3>

Gets or sets the name for custom report item to display in the widgets pane tile.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="category" data-Path="category-category">
<a href="#category" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>category</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the category name for the report item.

Defaults to *empty*

```
<h3 class="doc-prop-wrapper" id="tooltip" data-Path="tooltip-toolTip">
<a href="#tooltip" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>toolTip</span>
<span class="doc-prop-type"> object
</span>
</h3>
```

Gets information to provide content for custom report item tooltip.

| | | |
|--------------|--------------------------------------------------------------|--|
| Name | Description | |
| ----- ----- | | |
| requirements | Gets or sets the minimum values required for the report item | |
| description | Gets or sets the description content for the report item. | |
| title | Gets or sets the title for report item. | |

Defaults to *null*

Example

Add a custom report item

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
reportItemExtensions: [
name: 'barcode',
className: 'EJBarcode',
imageClass: 'customitem-barcode',
displayName: 'Barcode',
category: 'Custom ReportItem',
toolTip: {
requirements: 'Add a report item to the designer area.',
description: 'Display the barcode lines as report item.',
title: 'Barcode'
}
}]
});
</script>
`
```

Add multiple custom report items

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
reportItemExtensions: [
{
name: 'barcode',
className: 'EJBarcode',
imageClass: 'customitem-barcode',
displayName: 'Barcode',

```

toolbarSettings

Properties

```
category: 'Barcodes',
toolTip: {
  requirements: 'Add a report item to the designer area.',
  description: 'Display the barcode lines as report item.',
  title: 'Barcode'
}
},
{
name: 'qrbarcode',
className: 'EJQRBarcode',
imageClass: 'customitem-qrbarcode',
displayName: 'QR Barcode',
category: 'Barcodes',
toolTip: <!ItemTooltip>{
  requirements: 'Add a report item to the designer area.',
  description: 'Display the barcode lines as report item.',
  title: 'QR Barcode'
}
}]
});
});
</script>
`
```

toolbarSettings

Properties

```
<h3 class="doc-prop-wrapper" id="items" data-Path="items-items">
<a href="#items" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>items</span>
<span class="doc-prop-type"> enum
```

```
</span>
</h3>

<ts name = "ej.ReportDesigner.ToolbarItems" style = "display:none"></ts>
```

Shows or hides the grouped items in the toolbar with the help of `ej.ReportDesigner.ToolbarItems` enum.

| Name | Description |
|------------|--------------------------------------------------------------------------------------------------------------------------------------|
| New | Creates a new, blank report. |
| Open | Displays the Open dialog box to retrieve an existing report. |
| Save | Saves the active report to a specified location. |
| Cut | Removes the selected item from the active report. |
| Copy | Copies selected text or object to the Report Designer internal clipboard |
| Paste | Pastes the item that cut or copied into (the position of the insertion point) the report from the Report Designer internal clipboard |
| Delete | Deletes the selected item or text from the report. |
| Undo | Reverses the last action or deletes the last entry that is typed. |
| Redo | Reverses the action of the last Undo command. |
| Zoom | Used to "zoom in" to get a close-up view of a report or "zoom out" to see more of the page at a reduced size |
| Order | Used to change the layout order of report items in design area surface |
| Center | Aligns all report items to the center position of design surface in horizontal or vertical direction. |
| Alignment | Aligns the selected report item in the design surface |
| Distribute | Distributes selected report items at equal intervals from each other |
| Sizing | Equally size the selected report items in the design surface. |
| AlignGrid | Snaps the selected report items to the closest gridline. |
| EditDesign | Switches from preview to design view of the report. |
| View | Contains options to show or hide Header, Footer, Grid Lines, Snap To Shape in the report design. |
| Preview | Previews the active report in report viewer. |
| All | Shows all the toolbar items. |

Defaults to `ej.ReportDesigner.ToolbarItems.All`

```
<span>Examples</span>
```

Show all toolbar items

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
toolbarSettings: { items: ej.ReportDesigner.ToolbarItems.All }
});
</script>
'
```

Hide all toolbar items

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
toolbarSettings: { items: ~ej.ReportDesigner.ToolbarItems.All }
});
</script>
'
```

Hide **Zoom** option from toolbar

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
toolbarSettings: { items: ej.ReportDesigner.ToolbarItems.All & ~ej.ReportDesigner.ToolbarItems.Zoom }
});
</script>
'
```

Hide **Open** and **Save** option from toolbar

```
'html
<div id="container"></div>
```

```
<script>
$("#container").boldReportDesigner({
toolbarSettings: { items: ej.ReportDesigner.ToolbarItems.All & ~ej.ReportDesigner.ToolbarItems.Open &
~ej.ReportDesigner.ToolbarItems.Save }
});
</script>
`
```

```
<h3 class="doc-prop-wrapper" id="showtoolbar" data-Path="showtoolbar-showToolbar">
<a href="#showtoolbar" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>showToolbar</span>
```

```
<span class="doc-prop-type"> boolean
```

```
</span>
```

```
</h3>
```

Shows or hides the toolbar.

Defaults to *true*

```
<span>Examples</span>
```

Show report designer toolbar.

```
`html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
toolbarSettings: { showToolbar: true }
});
</script>
`
```

Hide report designer toolbar.

```
`html
```

pageSettings

Properties

```
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
toolbarSettings: { showToolbar: false }
});
</script>
` 

<h3 class="doc-prop-wrapper" id="templateid" data-Path="templateid-templateId">
<a href="#templateid" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>templateId</span>
<span class="doc-prop-type"> string
</span>
</h3>


Specifies the toolbar template ID.  
Defaults to empty


<span>Examples</span>
`html
<div id="container"></div>
<script>
$("#container").boldReportDesigner(
{
toolbarSettings:{templateId: "customtoolbarId"}
});
</script>
`
```

pageSettings

Properties

```
<h3 class="doc-prop-wrapper" id="orientation" data-Path="orientation-orientation">
```

```

<a href="#orientation" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>orientation</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.Orientation" style = "display:none"></ts>

```

Specifies the print layout orientation.

| Name | Description |
|-----------|---------------------------------------------------------------------------------------|
| Landscape | Specifies the Landscape property in pageSettings.orientation to get specified layout. |
| portrait | Specifies the portrait property in pageSettings.orientation to get specified layout. |

Defaults to *null*

```

<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
pageSettings:{ orientation: ej.ReportViewer.Orientation.Landscape }
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="pagesize" data-Path="pagesize-paperSize">
<a href="#pagesize" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>  
</a><span class='doc-prop-name'>paperSize</span>  
<span class="doc-prop-type"> enum  
</span>  
</h3>
```

```
<ts name = "ej.ReportViewer.PaperSize" style = "display:none"></ts>
```

Specifies the paper size of print layout.

| Name | Description |
|------------------|--------------------------------------------------------------------------------------|
| A3 | Specifies the A3 as value in pageSettings.paperSize to get specified size. |
| portrait | Specifies the A4 as value in pageSettings.paperSize to get specified size. |
| B4_JIS | Specifies the B4(JIS) as value in pageSettings.paperSize to get specified size. |
| B5_JIS | Specifies the B5(JIS) as value in pageSettings.paperSize to get specified size. |
| Envelope_10 | Specifies the Envelope #10 as value in pageSettings.paperSize to get specified size. |
| Envelope_Monarch | Specifies the Envelope as value in pageSettings.paperSize to get specified size. |
| Executive | Specifies the Executive as value in pageSettings.paperSize to get specified size. |
| Legal | Specifies the Legal as value in pageSettings.paperSize to get specified size. |
| Letter | Specifies the Letter as value in pageSettings.paperSize to get specified size. |
| Tabloid | Specifies the Tabloid as value in pageSettings.paperSize to get specified size. |
| Custom | Specifies the Custom as value in pageSettings.paperSize to get specified size. |

Defaults to *null*

```
<span style="font-weight:bold">Example</span>  
<js  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
    previewOptions: {  
        pageSettings:{ paperSize: ej.ReportViewer.PaperSize.A4 }  
    }  
});
```

```
</script>
`  
  
<h3 class="doc-prop-wrapper" id="height" data-Path="height-height">  
  <a href="#height" aria-hidden="true" class="anchor">  
    <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
      <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
    </svg>  
  </a><span class='doc-prop-name'>height</span>  
  <span class="doc-prop-type"> number  
  </span>  
</h3>  


Specifies the height of print layout.



Defaults to 0



<span style="font-weight:bold">Example</span>



```
'js
<div id="container"></div>
<script>
 $("#container").boldReportDesigner({
 previewOptions: {
 pageSettings: { height: 11.69 }
 }
 });
</script>
`

<h3 class="doc-prop-wrapper" id="width" data-Path="width-width">

 <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
 <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
 </svg>
```


```

```
</a><span class='doc-prop-name'>width</span>
<span class="doc-prop-type"> number
</span>
</h3>
```

Specifies the width of print layout.

Defaults to *0*

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
pageSettings: { width: 8.27 }
}
});
</script>
`
```

```
<h3 class="doc-prop-wrapper" id="margins" data-Path="margins-margins">
<a href="#margins" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>margins</span>
<span class="doc-prop-type"> number
</span>
</h3>
```

Specifies the margins of print layout.

| Name | Description |
|-------|----------------------------------------------|
| top | Specifies the top margins of print layout. |
| right | Specifies the right margins of print layout. |

previewOptions

Properties

| Name | Description |
|--------|-----------------------------------------------|
| bottom | Specifies the bottom margins of print layout. |
| left | Specifies the left margins of print layout. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
pageSettings: {
margins: { top: 0.5, right: 0.25, bottom: 0.25, left: 0.25 }
}
}
});
</script>
`
```

previewOptions

Properties

```
<h3 class="doc-prop-wrapper" id="datasources" data-Path="datasources-dataSources">
<a href="#datasources" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>dataSources</span>
<span class="doc-prop-type"> DataSources
<ts type = "array" style = "display:none"></ts>
</span>
</h3>
```

Gets or sets the list of data sources to preview RDL report from Report Designer.

```
<h3 class="doc-prop-wrapper" id="enableparameterblockscroller" data-Path="enableparameterblockscroller-enableParameterBlockScroller">
<a href="#enableparameterblockscroller" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>enableParameterBlockScroller</span>
<span class="doc-prop-type"> boolean
</span>
</h3>
Enables and disables the parameter block scroller in Report Viewer component .
Defaults to true
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
enableParameterBlockScroller: false
}
});
</script>
`


<h3 class="doc-prop-wrapper" id="enabledatasourceblockscroller" data-Path="enabledatasourceblockscroller-enableDatasourceBlockScroller">
<a href="#enabledatasourceblockscroller" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>enableDatasourceBlockScroller</span>
```

```
<span class="doc-prop-type"> boolean  
</span>  
</h3>
```

Enables and disables the data source credential block scroller.

Defaults to *true*

Example

```
'js  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
    previewOptions: {  
        enableDatasourceBlockScroller: false  
    }  
});  
</script>  
'
```

```
<h3 class="doc-prop-wrapper" id="enabledropdownsearch" data-Path="enabledropdownsearch-enableDropDownSearch">  
<a href="#enabledropdownsearch" aria-hidden="true" class="anchor">  
    <img alt="Icon representing dropdown search" data-bbox="111 558 883 643"/>  
</a><span class='doc-prop-name'>enableDropDownSearch</span>
```

```
<span class="doc-prop-type"> boolean  
</span>  
</h3>
```

Enables and disables the drop-down parameter search.

Defaults to *false*

Example

```
'js  
<div id="container"></div>
```

```
<script>
$("#container").boldReportDesigner({
previewOptions: {
enableDropDownSearch: true
}
});
</script>
`<h3 class="doc-prop-wrapper" id="exportsettings" data-Path="exportsettings-exportSettings">
<a href="#exportsettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>exportSettings</span>
<span class="doc-prop-type"> exportSettings
</span>
</h3>


Specifies the export settings for Report Viewer component.


<h3 class="doc-prop-wrapper" id="pagesettings" data-Path="pagesettings-pageSettings">
<a href="#pagesettings" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>pageSettings</span>
<span class="doc-prop-type"> pageSettings
</span>
</h3>


Specifies the page settings.


<h3 class="doc-prop-wrapper" id="parameters" data-Path="parameters-parameters">
```

```
<a href="#parameters" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>parameters</span>
```

```
<span class="doc-prop-type"> parameters
```

```
</span>
```

```
</h3>
```

Gets or sets the list of parameters associated with the report.

```
<h3 class="doc-prop-wrapper" id="parametersettings" data-Path="parametersettings-parameterSettings">
```

```
<a href="#parametersettings" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>parameterSettings</span>
```

```
<span class="doc-prop-type"> parameterSettings
```

```
</span>
```

```
</h3>
```

Specifies the parameter settings.

```
<h3 class="doc-prop-wrapper" id="printmode" data-Path="printmode-printMode">
```

```
<a href="#printmode" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>printMode</span>
```

```
<span class="doc-prop-type"> boolean
```

```
</span>
</h3>
```

Enables and disables the print mode.

Defaults to *false*

```
<span style="font-weight:bold">Example</span>
```

```
'js
```

```
<div id="container"></div>
```

```
<script>
```

```
$("#container").boldReportDesigner({
```

```
previewOptions: {
```

```
printMode:true
```

```
}
```

```
});
```

```
</script>
```

```
'
```

```
<h3 class="doc-prop-wrapper" id="printoption" data-Path="printoption-printOption">
<a href="#printoption" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>printOption</span>
```

```
<span class="doc-prop-type"> enum
```

```
</span>
```

```
</h3>
```

```
<ts name = "ej.ReportViewer.PrintOptions" style = "display:none"></ts>
```

Specifies the print option of the report.

| Name | Description |
|---------|-------------------------------------------------|
| Default | Specifies the Default property in printOptions. |
| NewTab | Specifies the NewTab property in printOptions. |
| None | Specifies the None property in printOptions. |

Defaults to `ej.ReportViewer.PrintOptions.Default`

Example

'js

<div id="container"></div>

<script>

`$("#container").boldReportDesigner({`

`previewOptions: {`

`printOption: ej.ReportViewer.PrintOptions.Default`

`}`

`});`

</script>

'

<h3 class="doc-prop-wrapper" id="toolbarsettings" data-Path="toolbarsettings-toolbarSettings">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

`<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>`

</svg>

toolbarSettings

 [toolbarSettings](#)

</h3>

Specifies the toolbar settings.

<h3 class="doc-prop-wrapper" id="zoomfactor" data-Path="zoomfactor-zoomFactor">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

`<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>`

</svg>

zoomFactor

 number

```
</span>
</h3>
Gets or sets the zoom factor for report viewer.
Defaults to 1
Example
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
zoomFactor: 2
}
});
</script>
`
```

Events

renderingBegin

Fires before report rendering is completed. If you want to perform any operation before the rendering of report,you can make use of the renderingBegin event.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
renderingBegin:function(args) {
// Write a code block to perform any operation before rendering.
}
}
```

```

    previewOptions
    renderingComplete

}

});

</script>
`
```

renderingComplete

Fires after report rendering completed. If you want to perform any operation after the rendering of report, you can make use of this renderingComplete event.

| Name | Type | Description |
|------------------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| model | object | returns the report model. |
| type | string | returns the name of the event. |
| reportParameters | object | returns the collection of parameters. |
| reportParameters | object | returns the collection of parameters. |

```

<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
renderingComplete:function(args) {
// Write a code block to perform any operation after rendering completed.
}
}
});
</script>
`
```

reportError

Fires when any error occurred while rendering the report. If you want to perform any operation when an error occurs in the report, you can make use of the reportError event.

| Name | Type | Description |
|--------|---------|---------------------------------------------------------|
| cancel | boolean | true if the event should be canceled; otherwise, false. |
| error | string | returns the error details. |

previewOptions

reportPrint

| Name | Type | Description |
|-------|--------|--------------------------------|
| model | object | returns the report model. |
| type | string | returns the name of the event. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
reportError: function (args) {
// Write a code block to perform any operation when report error occurs.
}
}
});
</script>
`
```

reportPrint

Fires when the report print action is performed in the report. To perform any operation during the report print action, use the ReportPrint event.

| Name | Type | Description |
|-------------|---------|---------------------------------------------------------------------|
| isStyleLoad | boolean | true if you have to load the external style file; otherwise, false. |

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
reportPrint: function (args) {
// Write a code block to perform any operation when report error occurs.
}
}
});
```

parameters Properties

});

</script>

'

parameters

Properties

```
<h3 class="doc-prop-wrapper" id="labels" data-Path="labels-labels">
<a href="#labels" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>labels</span>
```

```
<span class="doc-prop-type"> array
```

```
</span>
```

```
</h3>
```

Gets or sets the parameter labels.

Defaults to *null*

```
<span style="font-weight:bold">Example</span>
```

```
'js
```

```
<div id="container"></div>
```

```
<script>
```

```
$("#container").boldReportDesigner({
```

```
previewOptions: {
```

```
parameters: [
```

```
name:"Vehicle",
```

```
labels:["Motor Bikes"],
```

```
prompt:"Please select the color",
```

```
values:["Red","Green","Blue","Yellow","Black"],
```

```
nullable:false
```

```
}]
```

```
}
```

```
});
```

| | |
|------------|------------|
| parameters | Properties |
|------------|------------|

```

</script>
` 

<h3 class="doc-prop-wrapper" id="name" data-Path="name-name">
<a href="#name" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>name</span>
<span class="doc-prop-type"> string
</span>
</h3>

Gets or sets the name of the parameter.

Defaults to empty

<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
parameters: [{{
name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
}}]
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="nullable" data-Path="nullable-nullable">
```

```
<a href="#nullable" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>nullable</span>
<span class="doc-prop-type"> boolean
</span>
</h3>
Gets or sets whether the parameter allows nullable value or not.
Defaults to false
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
parameters: [{
name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
}]
}
});
</script>
`


<h3 class="doc-prop-wrapper" id="prompt" data-Path="prompt-prompt">
<a href="#prompt" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

</svg>

prompt

 string

</h3>

Gets or sets the prompt message associated with the specified parameter.

Defaults to *empty*

Example

`js

<div id="container"></div>

<script>

`($("#container").boldReportDesigner({`

`previewOptions: {`

`parameters: [{`

`name:"Vehicle",`

`labels:["Motor Bikes"],`

`prompt:"Please select the color",`

`values:["Red","Green","Blue","Yellow","Black"],`

`nullable:false`

`}]`

`}`

`});`

`</script>`

`,`

<h3 class="doc-prop-wrapper" id="values" data-Path="values-values">

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

exportSettings

Properties

```
</svg>
</a><span class='doc-prop-name'>values</span>
array
</span>
</h3>
Gets or sets the parameter values.
Defaults to []
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
parameters: [{{
name:"Vehicle",
labels:["Motor Bikes"],
prompt:"Please select the color",
values:["Red","Green","Blue","Yellow","Black"],
nullable:false
}}]
}
});
</script>
`
```

exportSettings

Properties

```
<h3 class="doc-prop-wrapper" id="exportoptions" data-Path="exportoptions-exportOptions">
<a href="#exportoptions" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
```

```
</a><span class='doc-prop-name'>exportOptions</span>
<span class="doc-prop-type"> enum
</span>
</h3>
```

<ts name = "ej.ReportViewer.ExportOptions" style = "display:none"></ts>

Specifies the export formats.

| Name | Description |
|-------------|--------------------------------------------------------------------------------|
| All | Specifies the All property in ExportOptions to get all available options. |
| Pdf | Specifies the Pdf property in ExportOptions to get Pdf option. |
| Word | Specifies the Word property in ExportOptions to get Word option. |
| Excel | Specifies the Excel property in ExportOptions to get Excel option. |
| Html | Specifies the Html property in ExportOptions to get Html option. |
| PPT | Specifies the PPT property in ExportOptions to get PPT option. |
| CSV | Specifies the CSV property in ExportOptions to get CSV option. |
| CustomItems | Specifies the customItems property in ExportOptions to get customItems option. |

Defaults to *ej.ReportViewer.ExportOptions.All*

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
exportSettings:{ exportOptions: ej.ReportViewer.ExportOptions.Html |
ej.ReportViewer.ExportOptions.Pdf }
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="excelformat" data-Path="excelformat-excelFormat">
<a href="#excelformat" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```

<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>excelFormat</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.ExcelFormats" style = "display:none"></ts>

```

Specifies the excel export format.

| Name | Description |
|---------------|---------------------------------------------------------------------------------------------------|
| Excel97to2003 | Specifies the Excel97to2003 property in ExcelFormats to get specified version of exported format. |
| Excel2007 | Specifies the Excel2007 property in ExcelFormats to get specified version of exported format. |
| Excel2010 | Specifies the Excel2010 property in ExcelFormats to get specified version of exported format. |
| Excel2013 | Specifies the Excel2013 property in ExcelFormats to get specified version of exported format. |

Defaults to *ej.ReportViewer.ExcelFormats.Excel97to2003*

```

<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
exportSettings:{ excelFormat: ej.ReportViewer.ExcelFormats.Excel97to2003}
}
});
</script>
`
```

```

<h3 class="doc-prop-wrapper" id="wordformat" data-Path="wordformat-wordFormat">
<a href="#wordformat" aria-hidden="true" class="anchor">
```

```

<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-
.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>wordFormat</span>
<span class="doc-prop-type"> enum
</span>
</h3>
<ts name = "ej.ReportViewer.WordFormats" style = "display:none"></ts>

```

Specifies the word export format.

| Name | Description |
|--------------|-------------------------------------------------------------------------------------------------|
| Doc | Specifies the Doc property in WordFormats to get specified version of exported format. |
| Dot | Specifies the Dot property in WordFormats to get specified version of exported format. |
| DOCX | Specifies the DOCX property in WordFormats to get specified version of exported format. |
| Word2007 | Specifies the Word2007 property in WordFormats to get specified version of exported format. |
| Word2010 | Specifies the Word2010 property in WordFormats to get specified version of exported format. |
| Word2013 | Specifies the Word2013 property in WordFormats to get specified version of exported format. |
| Word2007Dotx | Specifies the Word2007Dotx property in WordFormats to get specified version of exported format. |
| Word2010Dotx | Specifies the Word2010Dotx property in WordFormats to get specified version of exported format. |
| Word2013Dotx | Specifies the Word2013Dotx property in WordFormats to get specified version of exported format. |
| Word2007Docm | Specifies the Word2007Docm property in WordFormats to get specified version of exported format. |
| Word2010Docm | Specifies the Word2010Docm property in WordFormats to get specified version of exported format. |

| Name | Description |
|--------------|-------------------------------------------------------------------------------------------------|
| Word2013Docm | Specifies the Word2013Docm property in WordFormats to get specified version of exported format. |
| Word2007Dotm | Specifies the Word2007Dotm property in WordFormats to get specified version of exported format. |
| Word2010Dotm | Specifies the Word2010Dotm property in WordFormats to get specified version of exported format. |
| Word2013Dotm | Specifies the Word2013Dotm property in WordFormats to get specified version of exported format. |
| RTF | Specifies the RTF property in WordFormats to get specified version of exported format. |
| Txt | Specifies the Txt property in WordFormats to get specified version of exported format. |
| EPUB | Specifies the EPUB property in WordFormats to get specified version of exported format. |
| HTML | Specifies the HTML property in WordFormats to get specified version of exported format. |
| XML | Specifies the XML property in WordFormats to get specified version of exported format. |
| Automatic | Specifies the Automatic property in WordFormats to get specified version of exported format. |

Defaults to `ej.ReportViewer.WordFormats.Doc`

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
exportSettings:{ wordFormat: ej.ReportViewer.WordFormats.Doc}
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="customitems" data-Path="customitems-customItems">
```

```
<a href="#customitems" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>customItems</span>
<span class="doc-prop-type"> array
</span>
</h3>
Add the custom icon item to the export options.
Defaults to empty
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
  previewOptions: {
    exportSettings: {
      customItems: [{{
        index: 2,
        cssClass: "",
        value: 'exportCustom'
      },
      {
        index: 4,
        cssClass: "",
        value: 'exportCustom3'
      }]
    }
  }
});
</script>
```

toolbarSettings

Properties

```
<h3 class="doc-prop-wrapper" id="click" data-Path="click-click">
<a href="#click" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>click</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Fires when user click on toolbar item in the toolbar.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
toolbarSettings:{click: "onToolbarClick"}
}
});
</script>
`
```

```
<h3 class="doc-prop-wrapper" id="items" data-Path="items-items">
<a href="#items" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>items</span>
<span class="doc-prop-type"> object
</span>
</h3>
```

```
</svg>
</a><span class='doc-prop-name'>items</span>
enum
</span>
</h3>
<ts name = "ej.ReportViewer.ToolbarItems" style = "display:none"></ts>
```

Specifies the toolbar items.

| Name | Description |
|----------------|------------------------------------------------------------------------------|
| Print | Specifies the Print as value in ToolbarItems to get specified item. |
| Refresh | Specifies the Refresh as value in ToolbarItems to get specified item. |
| Stop | Specifies the Stop as value in ToolbarItems to get specified item. |
| Zoom | Specifies the Zoom as value in ToolbarItems to get specified item. |
| FitToPage | Specifies the FitToPage as value in ToolbarItems to get specified item. |
| Export | Specifies the Export as value in ToolbarItems to get specified item. |
| PageNavigation | Specifies the PageNavigation as value in ToolbarItems to get specified item. |
| Parameters | Specifies the Parameters as value in ToolbarItems to get specified item. |
| PrintLayout | Specifies the PrintLayout as value in ToolbarItems to get specified item. |
| PageSetup | Specifies the PageSetup as value in ToolbarItems to get specified item. |

Defaults to `ej.ReportViewer.ToolbarItems.All`

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
toolbarSettings:{ items: ej.ReportViewer.ToolbarItems.All }
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="showtoolbar" data-Path="showtoolbar-showToolbar">
```

```
<a href="#showtoolbar" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>showToolbar</span>
<span class="doc-prop-type"> boolean
</span>
</h3>
Shows or hides the toolbar.

Defaults to true

<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
  previewOptions: {
    toolbarSettings: { showToolbar: true }
  }
});
</script>
`



<h3 class="doc-prop-wrapper" id="showtooltip" data-Path="showtooltip-showTooltip">
<a href="#showtooltip" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>showTooltip</span>
<span class="doc-prop-type"> boolean
</span>
```

```
</h3>
```

Shows or hides the tooltip of toolbar items.

Defaults to *true*

```
<span style="font-weight:bold">Example</span>
```

```
`js
```

```
<div id="container"></div>
```

```
<script>
```

```
$("#container").boldReportDesigner({
```

```
previewOptions: {
```

```
toolbarSettings:{ showTooltip: true }
```

```
}
```

```
});
```

```
</script>
```

```
'
```

```
<h3 class="doc-prop-wrapper" id="templateid" data-Path="templateid-templateId">
```

```
<a href="#templateid" aria-hidden="true" class="anchor">
```

```
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>
```

```
</a><span class='doc-prop-name'>templateId</span>
```

```
<span class="doc-prop-type"> string
```

```
</span>
```

```
</h3>
```

Specifies the toolbar template ID.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
```

```
`js
```

```
<div id="container"></div>
```

```
<script>
```

```
$("#container").boldReportDesigner({
```

```
previewOptions: {
```

```
toolbarSettings:{ templateId: "customtoolbarId" }  
}  
});  
</script>  
  
<h3 class="doc-prop-wrapper" id="customitems" data-Path="customitems-customItems">  
<a href="#customitems" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
</svg>  
</a><span class='doc-prop-name'>customItems</span>  
<span class="doc-prop-type"> array  
</span>  
</h3>  
Add the custom icon item to the toolbar.  
Defaults to empty  
<span style="font-weight:bold">Example</span>  
'js  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
    previewOptions: {  
        toolbarSettings:{  
            customItems: [{  
                groupIndex: 1,  
                index: 1,  
                itemType: 'Default',  
                cssClass: "e-icon e-mail e-reportviewer-icon CustomItem",  
                tooltip: { header: 'CustomItem', content: 'toolbaritems', value: 'CustomItem' },  
            }]  
    }  
}
```

```
}

});

</script>

`



<h3 class="doc-prop-wrapper" id="customgroups" data-Path="customgroups-customGroups">
<a href="#customgroups" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>customGroups</span>
<span class="doc-prop-type"> array
</span>
</h3>
```

Add the custom icon groups to the toolbar.

Defaults to *empty*

```
<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
toolbarSettings:{
```

customGroups: [

```
items: [{
```

itemType: 'Default',

```
cssClass: "e-icon e-mail e-reportviewer-icon CustomGroup",
tooltip: { header: 'CustomGroup', content: 'toolbargroups', value: 'CustomGroup' },
},
```

{

```
itemType: 'Default',
cssClass: "e-icon e-mail e-reportviewer-icon subCustomGroup",
```

```

        tooltip: { header: 'subCustomGroup', content: 'subtoolbargroups', value: 'subCustomGroup' },
    }],
    groupIndex: 3
}],
}
}
});
</script>
`
```

parameterSettings

Properties

```

<h3 class="doc-prop-wrapper" id="delimiterchar" data-Path="delimiterchar-delimiterChar">
<a href="#delimiterchar" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 2.5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>delimiterChar</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Sets the separator when the multiSelectMode with delimiter option or checkbox is enabled with the dropdown. When you enter the delimiter value, the texts after the delimiter are considered as a separate word or query. The delimiter string is a single character and must be a symbol. Mostly, the delimiter symbol is used as comma (,) or semi-colon (;) or any other special character.

Defaults to ','

Example

`js

<div id="container"></div>

<script>

```

        $("#container").boldReportDesigner({
        previewOptions: {
        parameterSettings:{ delimiterChar: "," }`
```

```
}

});

</script>
`



<h3 class="doc-prop-wrapper" id="popupheight" data-Path="popupheight-popupHeight">
<a href="#popupheight" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>popupHeight</span>
<span class="doc-prop-type"> string
</span>
</h3>

Specifies the height of the combobox parameter popup list. By default, the popup height value is 152px.
Defaults to '152px'


<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
parameterSettings:{ popupHeight: "200px" }
}
});
</script>
`



<h3 class="doc-prop-wrapper" id="popupwidth" data-Path="popupwidth-popupWidth">
<a href="#popupwidth" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-
```

```
1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>  
</a><span class='doc-prop-name'>popupWidth</span>  
<span class="doc-prop-type"> string  
</span>  
</h3>
```

Specifies the width of the combobox parameter popup list. By default, the popup width sets based on the width of the component.

Defaults to '*auto*'

```
<span style="font-weight:bold">Example</span>
```

```
'js  
<div id="reportviewer"></div>
```

```
<script>  
$("#container").boldReportDesigner({  
    previewOptions: {  
        parameterSettings:{ popupWidth: "150px" }  
    }  
});  
</script>  
,
```

```
<h3 class="doc-prop-wrapper" id="itemwidth" data-Path="itemwidth-itemWidth">  
<a href="#itemwidth" aria-hidden="true" class="anchor">  
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

```
</svg>  
</a><span class='doc-prop-name'>itemWidth</span>  
<span class="doc-prop-type"> string  
</span>  
</h3>
```

Specifies the width of the parameter item. By default, the item width value is set as **185px**.

Defaults to '**185px**'

```
<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
parameterSettings:{ itemWidth: "150px" }
}
});
</script>
` 

<h3 class="doc-prop-wrapper" id="labelwidth" data-Path="labelwidth-labelWidth">
<a href="#labelwidth" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0 -.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>labelWidth</span>
<span class="doc-prop-type"> string
</span>
</h3>


Specifies the width of the parameter label. By default, the parameter label width value is set as 110px.  
Defaults to '110px'


<span style="font-weight:bold">Example</span>
`js
<div id="reportviewer"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
parameterSettings:{ labelWidth: "auto" }
}
});
</script>
```

dataSources `array` Properties

</script>

,

dataSources `array`

Properties

```
<h3 class="doc-prop-wrapper" id="name" data-Path="name-name">
<a href="#name" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>name</span>
<span class="doc-prop-type"> string
</span>
</h3>
```

Gets or sets the name of the data source.

Defaults to *empty*

Example

`js

<div id="container"></div>

<script>

```
$("#container").boldReportDesigner({
previewOptions: {
dataSources: [
name:"Menu Items",
value:[{ OrderId: "21D60", FoodName: "Burger", Price: 20, Category: "Veg" },
{ OrderId: "21D61", FoodName: "Pizza", Price: 25, Category: "Non-Veg" },
{ OrderId: "21D63", FoodName: "Sandwiches", Price: 30, Category: "Non-Veg" },
{ OrderId: "21D65", FoodName: "Chicken Drum Sticks", Price: 23, Category: "Non-Veg" },
{ OrderId: "21D64", FoodName: "Fulka", Price: 15, Category: "Veg" }
]]
}
});
```

```

</script>
` 

<h3 class="doc-prop-wrapper" id="value" data-Path="value-value">
<a href="#value" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
</svg>
</a><span class='doc-prop-name'>value</span>
<span class="doc-prop-type"> array
</span>
</h3>

Gets or sets the values of data source.

Defaults to [ ]

<span style="font-weight:bold">Example</span>
`js
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
previewOptions: {
dataSources: [
name:"Menu Items",
value:[{ OrderId: "21D60", FoodName: "Burger", Price: 20, Category: "Veg" },
{ OrderId: "21D61", FoodName: "Pizza", Price: 25, Category: "Non-Veg" },
{ OrderId: "21D63", FoodName: "Sandwiches", Price: 30, Category: "Non-Veg" },
{ OrderId: "21D65", FoodName: "Chicken Drum Sticks", Price: 23, Category: "Non-Veg" },
{ OrderId: "21D64", FoodName: "Fulka", Price: 15, Category: "Veg" }
]]
}
});
</script>
` 

```

Available chart types

| | | |
|-----------------------|-----------------------------|--|
| Chart Types | Input Text | |
| ----- ----- | | |
| Column | Chart-Column | |
| StackedColumn | Chart-StackedColumn | |
| StackedColumnPercent | Chart-StackedColumnPercent | |
| Bar | Chart-Bar | |
| StackedBar | Chart-StackedBar | |
| StackedBarPercent | Chart-StackedBarPercent | |
| Pie | Chart-Pie | |
| ExplodedPie | Chart-ExplodedPie | |
| Doughnut | Chart-Doughnut | |
| Pyramid | Chart-Pyramid | |
| Funnel | Chart-Funnel | |
| Area | Chart-Area | |
| SmoothArea | Chart-SmoothArea | |
| StackedArea | Chart-StackedArea | |
| StackedAreaPercent | Chart-StackedAreaPercent | |
| Line | Chart-Line | |
| SmoothLine | Chart-SmoothLine | |
| SteppedLine | Chart-SteppedLine | |
| LineWithMarkers | Chart-LineWithMarkers | |
| SmoothLineWithMarkers | Chart-SmoothLineWithMarkers | |
| Bubble | Chart-Bubble | |
| Radar | Chart-Radar | |
| Polar | Chart-Polar | |
| Scatter | Chart-Scatter | |

permissionSettings

Properties

```
<h3 class="doc-prop-wrapper" id="dataset" data-Path="dataset-dataSet">
<a href="#dataset" aria-hidden="true" class="anchor">
<svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">
```

```
<path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-.91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2 1.22-2 2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2 3.25C6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>
```

</svg>

dataSet

 enum

</h3>

<ts name = "ej.ReportDesigner.Permission" style = "display:none"></ts>

Shows or hides the create, edit and delete options in dataset pane with the help of ej.ReportDesigner.Permission enum.

| Name | Description |
|--------|--------------------------------------------------|
| ----- | ----- |
| Create | Shows or hides create option in dataset pane |
| Edit | Shows or hides the edit option in dataset pane |
| Delete | Shows or hides the delete option in dataset pane |
| All | Shows all the options in dataset pane |

Defaults to *ej.ReportDesigner.Permission.All*

Example

Show all options in dataset pane.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
    permissionSettings: { dataSet: ej.ReportDesigner.Permission.All }
});
</script>
`
```

Hide all options in dataset pane.

```
'html
<div id="container"></div>
<script>
```

```
$("#container").boldReportDesigner({  
    permissionSettings: { dataSet: ej.ReportDesigner.Permission.All }  
});  
</script>  
\
```

Hide **Create** option from dataset pane.

```
'html  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
    permissionSettings: { dataSet: ej.ReportDesigner.Permission.All &  
        ~ej.ReportDesigner.Permission.Create}  
});  
</script>  
\
```

Hide all items except **Create** option from dataset pane.

```
'html  
<div id="container"></div>  
<script>  
$("#container").boldReportDesigner({  
    permissionSettings: { dataSet: ej.ReportDesigner.Permission.All & ~ej.ReportDesigner.Permission.Edit &  
        ~ej.ReportDesigner.Permission.Delete}  
});  
</script>  
\  
  
<h3 class="doc-prop-wrapper" id="datasource" data-Path="datasource-dataSource">  
    <a href="#datasource" aria-hidden="true" class="anchor">  
        <svg aria-hidden="true" height="16" version="1.1" viewBox="0 0 16 16" width="16">  
            <path fill-rule="evenodd" d="M4 9h1v1H4c-1.5 0-3-1.69-3-3.5S2.55 3 4 3h4c1.45 0 3 1.69 3 3.5 0 1.41-  
            .91 .72-2 .25V8.59c.58-.45 1-1.27 1-2.09C10 5.22 8.98 4 8 4H4c-.98 0-2 1.22-2 2.5S3 9 4 9zm9-3h-  
            1v1h1c1 0 2 1.22 2 .5S13.98 12 13 12H9c-.98 0-2-1.22-2-2.5 0-.83.42-1.64 1-2.09V6.25c-1.09.53-2 1.84-2  
            3.25S6 11.31 7.55 13 9 3h4c1.45 0 3-1.69 3-3.5S14.5 6 13 6z"></path>  
        </svg>  
    </a>  
</h3>
```

```
</a><span class='doc-prop-name'>dataSource</span>
<span class="doc-prop-type"> enum
</span>
</h3>
```

`<ts name = "ej.ReportDesigner.Permission" style = "display:none"></ts>`
Shows or hides the create, edit and delete options in data source pane with the help of `ej.ReportDesigner.Permission` enum

| Name | Description |
|--------|------------------------------------------------------|
| Create | Shows or hides create option in data source pane |
| Edit | Shows or hides the edit option in data source pane |
| Delete | Shows or hides the delete option in data source pane |
| All | Shows all the options in data source pane |

Defaults to `ej.ReportDesigner.Permission.All`

Example

Show all options in data source pane.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
    permissionSettings: { dataSource: ej.ReportDesigner.Permission.All }
});
</script>
`
```

Hide all options in data source pane.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
    permissionSettings: { dataSource: ~ej.ReportDesigner.Permission.All }
});
</script>
`
```

Hide **Create** option from data source pane.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
    permissionSettings: { dataSource: ej.ReportDesigner.Permission.All &
~ej.ReportDesigner.Permission.Create}
});
</script>
`
```

Hide all items except **Create** option from data source pane.

```
'html
<div id="container"></div>
<script>
$("#container").boldReportDesigner({
    permissionSettings: { dataSource: ej.ReportDesigner.Permission.All &
~ej.ReportDesigner.Permission.Edit & ~ej.ReportDesigner.Permission.Delete}
});
</script>
`
```

Bold reporting tools for React

Enterprise-class reporting tools for React development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your React applications.

How to best read this user guide

The best way to get started would be to read the **Getting Started** section of the documentation for the control that you would like to start using first. The **Getting Started** guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Bold reporting tools for React

Enterprise-class reporting tools for React development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your React applications.

How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

System Requirements

This topic describes the software requirements for setting up the development environment of Bold Reports React.

Supported Operating Systems

Windows 7+, 8+

Windows Server 2008 R2+

Software Requirements

The following software requirements are necessary for setting up the development environment of Bold Reports React:

Microsoft Visual Studio Code

[Node JS](#) (version 8.x or 10.x)

[NPM](#) (v3.x.x or higher)

Browser Compatibility

IE 9+

Microsoft Edge

Mozilla Firefox 22+

Chrome 17+

Opera 12+

Safari 5+

See Also

[Licensing procedure for deployment](#)

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Excel, CSV, Word, PowerPoint, and HTML formats. Some of the key features are:

Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.

Easily customize each element of the Report Viewer and provide events for report processing customization.

Supports jQuery, Angular, React, Ember, Aurelia, PHP, and JSP.

Add Web Report Viewer to a React application

This section explains the steps required to add web Report Viewer to a React application.

Prerequisites

Before getting started with the bold report viewer, make sure that your development environment includes the following commands.

[Node JS \(version 8.x or 10.x\)](#)

[NPM \(v3.x.x or higher\)](#)

Install the Create React App package

Create React App is a simple way to create single-page React application which provides a build setup with no configuration. To install the Create React App globally in your machine, run the following command in Command Prompt.

```
'typescript  
npm install create-react-app -g  
'
```

To learn more about Create React App package refer [here](#)

Create a new React application

To create a new React application, run the following command in Command Prompt.

```
'typescript  
create-react-app reports  
'
```

The `create-react-app` command adds the `react`, `react-dom`, `react-scripts`, and other dependencies required to your React application.

Install the Create React Class

To configure the Report Viewer component, change the directory to your application's root folder.

```
'typescript
```

```
cd reports
```

```
'
```

To install the type definitions for `create-react-class`, run the following command in Command Prompt.

```
'typescript
```

```
npm install create-react-class --save
```

```
'
```

Install the Bold Reports React package

To install the Bold Reports React package, run the following command in Command Prompt.

```
'typescript
```

```
npm install @boldreports/react-reporting-components --save
```

```
'
```

Adding scripts reference

Bold Reports needs `window.jQuery` object to render the React components. Hence create a file named `globals.js` and import `jQuery` in `globals.js` file as like the below code snippet.

```
'typescript
```

```
import jquery from 'jquery';
import React from 'react';
import createReactClass from 'create-react-class';
import ReactDOM from 'react-dom';
window.React = React;
window.createReactClass = createReactClass;
windowReactDOM = ReactDOM;
window.$ = window.jQuery = jquery;
```

```
'
```

Refer the `globals.js` file in `index.js` file as like below code snippet.

```
'typescript
```

```
import './globals'
import React from 'react';
import ReactDOM from 'react-dom';
```

```
import './index.css';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
`
```

Adding Report Viewer component

The Bold Report Viewer script and style files need to be imported, in order to run the web Report Viewer. Hence, import the following files in `App.js` file.

```
'typescript
/*eslint-disable */
import React from 'react';
import './App.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
var viewerStyle = {'height': '700px', 'width': '100%'};
function App() {
  return (
    <div style={viewerStyle}>
      <BoldReportViewerComponent
        id="reportviewer-container">
      </BoldReportViewerComponent>
    </div>
  );
}
export default App;
`
```

Open the `public\index.html` and refer the following scripts in `<head>` tag.

```
'html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
'
```

Create a Web API service

The web Report Viewer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set a Web API service URL

To set a Web API service, open the `App.js` file and replace the existing code with the below code snippet.

```
'javascript
/*eslint-disable */
import React from 'react';
import './App.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
```

```
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
var viewerStyle = {'height': '700px', 'width': '100%'};
function App() {
  return (
    <div style={viewerStyle}>
      <BoldReportViewerComponent
        id="reportviewer-container"
        reportServiceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}
        reportPath={'~/Resources/docs/sales-order-detail.rdl'}>
      </BoldReportViewerComponent>
    </div>
  );
}
export default App;
```

Run the Application

To run the application, run the following command in command prompt.

```
'typescript
npm run start
'
```

The `npm run start` command automatically opens your browser to `http://localhost:3000/[report viewer output]/static/assets/react/report-viewer/images/Getting-Started_img1.png`

Deploying the application in production

To deploy the application in production, run the following command in command prompt which will create a folder named `build` to generate the build files.

```
'typescript
npm run build
'
```

Add Web Report Viewer with Typescript React application

This section explains the steps required to add web Report Viewer to a React Typescript application.

Prerequisites

Before getting started with bold web report viewer, make sure your development environment includes the following,

[Node JS \(version 8.x or 10.x\)](#)[NPM \(v3.x.x or higher\)](#)[Install the Create React App Package](#)

Create React App is a simple way to create single-page React application which provides a build setup with no configuration. To install the Create React App globally in your machine, run the following command in Command Prompt.

```
'typescript  
npm install create-react-app -g  
'
```

To learn more about Create React App package refer [here](#)

[Create a new React Typescript Application](#)

To create a new React typescript application, run the below command in Command Prompt.

```
'typescript  
create-react-app reports --template typescript  
'
```

The `create-react-app` command adds the `react`, `react-dom`, `react-scripts` and other dependencies required to your react typescript application.

[Install Create React Class](#)

To configure the Report Viewer component, change the directory to your application's root folder.

```
'typescript  
cd reports  
'
```

To install the type definitions for `create-react-class` run the following command in Command Prompt.

```
'typescript  
npm install create-react-class --save  
npm install @types/create-react-class --save  
'
```

[Install Bold Reports React package](#)

To install the Bold Reports React package run the following command in Command Prompt.

```
'typescript  
npm install @boldreports/react-reporting-components --save  
'
```

[Install JQuery package](#)

To install the JQuery package run the following command in Command Prompt.

```
'typescript
npm install @types/jquery --save
'
```

[Adding Scripts reference](#)

Bold Reports needs `window.jQuery` object to render the React components. Hence create a file named `globals.ts` and import `jQuery` in `globals.ts` file as like the below code snippet.

```
'typescript
import jquery from 'jquery';
import React from 'react';
import createReactClass from 'create-react-class';
import ReactDOM from 'react-dom';
(window as any).React = React;
(window as any).createReactClass = createReactClass;
(window as any).ReactDOM = ReactDOM;
(window as any).$ = (window as any).jQuery = jquery;
'
```

Refer the `globals.ts` file in `index.tsx` file as like below code snippet.

```
'typescript
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
'
```

[Adding Report Viewer component](#)

The web Report Viewer script and style files need to be imported in order for the web Report Viewer to run. Hence, import the following files in `App.tsx` file.

```
'typescript
/*eslint-disable */
import React from 'react';
import './App.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
```

```

import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization

import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';

//Reports react base

import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';

declare let BoldReportViewerComponent: any;

var viewerStyle = {

  'height': '700px',
  'width': '100%'

};

function App() {

  return (

    <div style={viewerStyle}>

      <BoldReportViewerComponent
        id="reportviewer-container">
      </BoldReportViewerComponent>

    </div>
  );
}

export default App;
`
```

Open the `public\index.html` and refer the following scripts in `<head>` tag.

```

`html
<!-- Data-Visualization -->

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js"></script>
```

```
.
```

Create a Web API service

The web Report Viewer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set Web API service URL

To set Web API service, open the `app.component.ts` file and add the code snippet as in the constructor.

```
'javascript
/eslint-disable /
import React from 'react';
import './App.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
declare let BoldReportViewerComponent: any;
var viewerStyle = {
  'height': '700px',
  'width': '100%'
};
```

```
function App() {  
  return (  
    <div style={viewerStyle}>  
      <BoldReportViewerComponent  
        id="reportviewer-container"  
        reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}  
        reportPath = {'~/Resources/docs/sales-order-detail.rdl'} >  
    </BoldReportViewerComponent>  
  </div>  
);  
}  
  
export default App;
```

Run the Application

To run the application, run the following command in command prompt

```
'typescript  
npm run start  
'
```

The `npm run start` command automatically opens your browser to <http://localhost:3000/>![report viewer output](/static/assets/react/report-viewer/images/Getting-Started_img1.png)

Deploying the application in production

To deploy the application in production we need to generate the build files. Hence to generate the build files run the below command in command prompt which will create a folder named `build`

```
'typescript  
npm run build  
'
```

Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports, set the `reportServerUrl`, `reportPath` and `reportServiceUrl` properties as shown in the following code snippet in `App.js` or `app.component.ts` file.

```
'javascript  
function App() {  
  return (<div id="viewer" style={viewerStyle}>  
    <BoldReportViewerComponent id="reportviewer-container"
```

```
reportServiceUrl = {'https://demos.boldreports.com/services/api/Report Viewer'}
reportPath = {'/SSRSSamples2/Territory Sales_Link'}
reportServerUrl = {'http://<servername>/reportserver$instanceName'}
```

```
</BoldReportViewerComponent>
</div>);
}
`
```

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of `/folder name/report name`.

Network credentials for SSRS

The network credentials are required to connect with the specified SSRS Report Server using the Report Viewer. Specify the `ReportServerCredential` property in the Web API Controller `OnInitReportOptions` method.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server credential
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
}
```

Set data source credential for shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS server. If the report has any data source that uses credentials to connect with the database, then you should specify the `DataSourceCredentials` for each report data source to establish database connection.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-demo",
    "N@c)=Y8s*1&dh"));
```

```
}
```

Data source credentials should be added to the shared data sources that do not have credentials in the connection strings.

Build and run the application.

Change data source connection string

You can change the connection string of a report data source before it is loaded in the Report Viewer.

The **DataSourceCredentials** class provides the option to set and update the modified connection string as in the following code snippet.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.DataSourceCredentials.Add(new
        BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-
        demo","N@c)=Y8s*1&dh","Data Source=dataplatformdemodata.syncfusion.com;Initial
        Catalog=AdventureWorks"));
}
```

The previous code shows an option to change the connection string only, but the class provides multiple options to change data source information. To learn more about this, refer to the **DataSourceCredentials** class.

Render linked reports

You can render a linked report that points to an existing report, which is published in the SSRS Report Server. You can also set the parameter, data source, credential, and other properties like normal SSRS reports using the Report Viewer.

```
'javascript
```

```
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/Report Viewer'}
            reportPath = {'/SSRSSamples2/Territory Sales_Link'}
            reportServerUrl = {'http://<servername>/reportserver$instanceName'}
        </BoldReportViewerComponent>
    </div>);
}
```

The **Territory Sales_Link** is a linked report created for the **Territory Sales** report, which is already available in the SSRS Report Server.

Add Web Report Viewer to a React Boilerplate application

This section explains the steps required to add web [Bold Report Viewer](#) to [React Boilerplate application](#).

Prerequisites

Before getting started with the report viewer, make sure that you have the following requirements.

[Node JS](#) (version 8.x or 10.x)

[NPM](#) (v3.x.x or higher)

To quick start with React Boilerplate application, we have already configured our [Bold Reports with React Boilerplate application](#). Those who are wish to directly getting started with Bold Reports React Boilerplate application execute the below commands.

```
'bash  
git clone https://github.com/boldreports/react-boilerplate.git  
cd react-boilerplate  
npm install  
npm start  
'
```

React Boilerplate Application

Download React Boilerplate application from this [link](#) and extract it.

From the extracted folder, execute the following command to install project dependencies.

```
'bash  
npm install  
'
```

Install the Bold Reports React package

To install the [Bold Reports React](#) package, run the following command in command Prompt.

```
'bash  
npm install @boldreports/react-reporting-components --save-dev  
'
```

Also, install [create-react-class](#) package, which is required by Bold Reports React package.

```
'bash
```

```
npm install create-react-class --save-dev
```

Adding global scripts reference

Since, Bold Reports requires global jquery, React, createReactClass and ReactDOM objects to render the Report components, we need to import and assign those to window object in a newly created app/globals.js file.

```
'typescript
import jquery from 'jquery';
import React from 'react';
import createReactClass from 'create-react-class';
import ReactDOM from 'react-dom';
window.React = React;
window.createReactClass = createReactClass;
window.ReactDOM = ReactDOM;
window.$ = window.jQuery = jquery;
'
```

Import the globals.js file in app/app.js file as like below code snippet.

```
'javascript
/
app.js *
```

This is the entry file for the application, only setup and boilerplate code.

```
*/
import './globals';
// Needed for redux-saga es6 generator support
import '@babel/polyfill';
// Import all the third party stuff
....
```

Configuring webpack

Since, Bold Reports uses `cur` type files, we need to provide support to load such type of file in webpack `url-loader` plugin by configuring `internals/webpack/webpack.base.babel.js` file.

```
'javascript
.....
....
test: /\.jpg|png|gif|cur$/,
use: [
{
loader: 'url-loader',
options: {
// Inline files smaller than 10 kB
limit: 10 * 1024,
},
}
.....
.....
`'
```

Adding Report Viewer component

The Bold Report Viewer `script` and `style` files are needs to be imported in order to run the web viewer.

So, import the following scripts and css in `app/app.js` file. Now, use the tag

`BoldReportViewerComponent` to render our viewer in the application.

```
'javascript
.....
....
import history from 'utils/history';
import 'sanitize.css/sanitize.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
```

```
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';

....
....
const MOUNT_NODE = document.getElementById('app');

const bounds = { height: '800px', width: '100%'};

const render = messages => {
  ReactDOM.render(
    <div style={bounds}>
      <BoldReportViewerComponent id="reportviewer-container" >
      </BoldReportViewerComponent>
    </div>,
    MOUNT_NODE,
  );
};

....
....
`
```

Open the `app\index.html` and refer the following scripts in `<head>` tag.

```
'html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
`
```

Create a Web API service

The web Report Viewer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)[ASP.NET Core Web API Service](#)

Set a Web API service URL

Bind an online reportServiceUrl to Bold Report viewer component in app/app.js file like the below snippet.

```
'javascript
.....
....
import history from 'utils/history';
import 'sanitize.css/sanitize.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
.....
....
const MOUNT_NODE = document.getElementById('app');
const bounds = { height: '800px', width: '100%'};
const render = messages => {
  ReactDOM.render(
    <div style={bounds}>
      <BoldReportViewerComponent
        id="reportviewer-container"
        reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
        reportPath = {'~/Resources/docs/sales-order-detail.rdl'} >
        </BoldReportViewerComponent>
    </div>,
  MOUNT_NODE,
```

```
);
```

```
};
```

```
....
```

```
....
```

```
\
```

Run the Application

To run the app, execute the following command and browse to <http://localhost:3000> to see the application.

```
`bash
```

```
npm start
```

```
\
```

```
![report viewer output](/static/assets/react/report-viewer/images/react-boilerplate.png)
```

Deploying the application in production

To deploy the application in production, run the following command in command prompt which will create a folder named `build` to generate the build files.

```
`bash
```

```
npm run build
```

```
\
```

Add Web Report Viewer to a React Boilerplate application

This section explains the steps required to add web [Bold Report Viewer](#) to [React Boilerplate TypeScript application](#).

Prerequisites

Before getting started with the report viewer, make sure that you have the following requirements.

[Node JS](#) (version 8.x or 10.x)

[NPM](#) (v3.x.x or higher)

To quick start with React Boilerplate application, we have already configured our [Bold Reports with React Boilerplate TypeScript application](#). Those who are wish to directly getting started with Bold Reports React Boilerplate TypeScript application execute the below commands.

```
`bash
```

```
git clone https://github.com/boldreports/react-boilerplate-typescript.git
```

```
cd react-boilerplate
```

```
npm install
```

```
npm start
```

React Boilerplate TypeScript Application

Download React Boilerplate application from this [link](#) and extract it.

From the extracted folder, execute the following command to install project dependencies.

```
'bash  
npm install  
'
```

Install the Bold Reports React package

To install the [Bold Reports React](#) package, run the following command in command Prompt.

```
'bash  
npm install @boldreports/react-reporting-components --save-dev  
'
```

Also, install [create-react-class](#) package, which is required by Bold Reports React package.

```
'bash  
npm install create-react-class --save-dev  
'
```

Adding global scripts reference

Since, Bold Reports requires global `jquery`, `React`, `createClass` and `ReactDOM` objects to render the Report components, we need to import and assign those to `window` object in a newly created `app/globals.ts` file.

```
'typescript  
import jquery from 'jquery';  
import React from 'react';  
import createReactClass from 'create-react-class';  
import ReactDOM from 'react-dom';  
(window as any).React = React;  
(window as any).createClass = createReactClass;  
(window as any).ReactDOM = ReactDOM;  
(window as any).$ = (window as any).jQuery = jquery;  
'
```

Import the `globals.ts` file in `app/app.tsx` file as like below code snippet.

```
`javascript
/
  app.tsx *

This is the entry file for the application, only setup and boilerplate
code.

*/
import './globals';
// Needed for redux-saga es6 generator support
import 'react-app-polyfill/ie11';
import 'react-app-polyfill/stable';
// Import all the third party stuff
....
....
`
```

Configuring webpack

Since, Bold Reports uses `cur` type files, we need to provide support to load such type of file in webpack `url-loader` plugin by configuring `internals/webpack/webpack.base.babel.js` file.

```
`javascript
....
....
test: /\.jpg|png|gif|cur$/,
use: [
{
  loader: 'url-loader',
  options: {
    // Inline files smaller than 10 kB
    limit: 10 * 1024,
  },
}
....
....
```

\

Also, change the image quality in `image-webpack-loader` from `65-90` to `[0.65, 0.90]`.

`javascript

....

....

{

loader: 'image-webpack-loader',

options: {

mozjpeg: {

enabled: false,

// NOTE: mozjpeg is disabled as it causes errors in some Linux environments

// Try enabling it in your environment by switching the config to:

// enabled: true,

// progressive: true,

},

gifsicle: {

interlaced: false,

},

optipng: {

optimizationLevel: 7,

},

pngquant: {

quality: [0.65, 0.90],

speed: 4,

},

},

}

....

....

\

Adding Report Viewer component

The Bold Report Designer script and style files are needs to be imported in order to run the web viewer. So, import the following scripts and css in app/app.tsx file. Now, use the tag BoldReportViewerComponent to render our viewer in the application.

'javascript

....

....

```
import history from 'utils/history';
import 'sanitize.css/sanitize.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
.....
....
declare let BoldReportViewerComponent: any;
const MOUNT_NODE = document.getElementById('app') as HTMLElement;
const bounds = { height: '800px', width: '100%'};
const ConnectedApp = (props: { messages: any }) => (
<div style={bounds}>
<BoldReportViewerComponent id="reportviewer-container" >
</BoldReportViewerComponent>
</div>
);
const render = (messages: any) => {
ReactDOM.render(<ConnectedApp messages={messages} />, MOUNT_NODE);
};
.....
....
```

Open the `app\index.html` and refer the following scripts in `<head>` tag.

```
'html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
'
```

Create a Web API service

The web Report Viewer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set a Web API service URL

Bind an online `reportServiceUrl` to Bold Report viewer component in `app/app.tsx` file like the below snippet.

```
'javascript
....
....
import history from 'utils/history';
import 'sanitize.css/sanitize.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
```

```
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
.....
.....
declare let BoldReportViewerComponent: any;
const MOUNT_NODE = document.getElementById('app') as HTMLElement;
const bounds = { height: '800px', width: '100%'};
const ConnectedApp = (props: { messages: any }) => (
<div style={bounds}>
<BoldReportViewerComponent
id="reportviewer-container"
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = {'~/Resources/docs/sales-order-detail.rdl'} >
</BoldReportViewerComponent>
</div>
);
const render = (messages: any) => {
ReactDOM.render(<ConnectedApp messages={messages} />, MOUNT_NODE);
};
.....
.....
`
```

Run the Application

To run the app, execute the following command and browse to <http://localhost:3000> to see the application.

```
'bash
npm start
`
```

![report viewer output](/static/assets/react/report-viewer/images/react-boilerplate.png)

Deploying the application in production

To deploy the application in production, run the following command in command prompt which will create a folder named build to generate the build files.

```
'bash
npm run build
```

Load Bold Reports Report Server reports

This section explains how to render the Bold Reports Report Server reports in your React application. The Bold Reports Report Server contains built-in Web API service that allows you to display the reports in your application without creating a new web API. You need a React application configured with Bold Reports resources to do the following steps:

If you don't have React application with configured Bold Reports resource, then create the app using the [Getting-started](#) steps.

The Report Viewer requires the `serviceAuthorizationToken` for report service authorization.

Create a token for the user by using the Bold Reports Report Server RESTful API. Replace the following code snippet in `index.js` file to generate the token.

```
'js
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
const $ = window.$;
$(function () {
  var dataValue = "";
  var apiRequest ={userid : 'guest@boldreports.com',password : 'demo' };
  $.ajax({
    type: "POST",
    url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
    data: apiRequest,
    success: function (data) {
      dataValue = data.Token;
      var token = JSON.parse(dataValue);
      // Report Viewer initialization.
    }
  });
});`
```

You need the `serviceAuthorizationToken` as mandatory for Report Viewer to interact with Bold Reports Report Server Service. So, pass the generated token to `App`, in order to set the `serviceAuthorizationToken` for Report Viewer. Use the following code snippet in `index.js` file to send the token to `App`.

```
'js
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
const $ = window.$;
$(function () {
  var dataValue = "";
  var apiRequest ={userid : 'guest@boldreports.com',password : 'demo' };
  $.ajax({
    type: "POST",
    url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
    data: apiRequest,
    success: function (data) {
      dataValue = data.Token;
      var token = JSON.parse(dataValue);
      ReactDOM.render(<App {...token}></App>, document.getElementById('root'));
    }
  });
});
});'
`
```

Open the `App.js` file and add token parameter with `App` function to get the `serviceAuthorizationToken` for an application page and set `reportPath`, `serviceAuthorizationToken`, and `reportServiceUrl` for Report Viewer. Also, you have to provide the `serverurl` with `ajaxBeforeLoad` as like in the following code example.

```
'js
function App(token) {
  return (<div id="viewer" style={viewerStyle}>
```

Render RDLC report

Bind data source at client side

```
<BoldReportViewerComponent id="reportviewer-container"
reportServiceUrl = {'https://on-premise-demo.boldreports.com/reporting/reportservice/api/Viewer'}
reportPath = {'/Sample Reports/Product Line Sales'}
ajaxBeforeLoad = {onAjaxRequest}
serviceAuthorizationToken = {token["tokentype"] + " " + token["accesstoken"]}

</BoldReportViewerComponent>
</div>);
}

function onAjaxRequest(args) {
args.headers.push({
Key: 'serverurl', Value: 'https://on-premise-demo.boldreports.com/reporting/api/site/site1'
});
}

`
```

Navigate to the root of the application and run using the following command.

```
`typescript
npm run start
`
```

Render RDLC report

The data binding support allows you view the RDLC reports that exist on the local file system with JSON array and custom business object data collection. The following steps demonstrates how to render an RDLC report with JSON array and custom business object data collection.

Add the RDLC report **Product List.rdlc** from the Bold Reports installation location to your application **App_Data** folder. For more information, refer to [Samples and demos](#).

Bind data source at client side

To bind the data source at client side, follow these steps;

Set the RDLC report path to the **reportPath** property.

Assign the **processingMode** property to **ProcessingMode.Local**.

Bind the JSON array collection to the **dataSources** property as shown in following code.

```
`javascript
```

```
var viewerStyle = {'height': '700px', 'width': '100%'};

var reportPath = 'AreaCharts.rdlc';

var reportData = [{

value: [

{ SalesPersonID: 281, FullName: 'Ito', Title: 'Sales Representative', SalesTerritory: 'South West', Y2002: 0, Y2003: 28000, Y2004: 3018725 },

{ SalesPersonID: 282, FullName: 'Saraiva', Title: 'Sales Representative', SalesTerritory: 'Canada', Y2002: 25000, Y2003: 14000, Y2004: 3189356 },

{ SalesPersonID: 283, FullName: 'Cambell', Title: 'Sales Representative', SalesTerritory: 'North West', Y2002: 12000, Y2003: 13000, Y2004: 1930885 }

],



name: 'AdventureWorksXMLDataSet'

}];

function App() {

return (

<div style={viewerStyle}>

<BoldReportViewerComponent

id="reportviewer-container"

reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}

reportPath = {'~/Resources/docs/sales-order-detail.rdl'}

processingMode = {"Local"}

reportPath = {reportPath}>

</BoldReportViewerComponent>

</div>

);

}

export default App;
```

Build and run the application.

Bind data source in Web API controller

The following steps help you configure the Web API to render the RDLC report with business object data collection.

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
`csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList<ProductList> GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
            Category = "Non-Veg",
            Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
            ProductImage = ""
        };
        datas.Add(data);
    }
}
```

Render RDLC report

Load report as stream

```
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};

datas.Add(data);
return datas;
}
```

Set the value of the **ProcessingMode** property to **ProcessingMode.Local** and **ReportPath** in the RDLC report location.

Bind the business object data values collection by adding a new item to the **DataSources** as shown in the following code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;
    reportOption.ReportModel.ReportPath =
        System.Web.Hosting.HostingEnvironment.MapPath("~/App_Data/Product List.rdlc");
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
```

Here the **Name** is case sensitive and it should be same as in the data source name in the report definition.

The **Value** accepts **IList**, **DataSet**, and **DataTable** inputs.

[Load report as stream](#)

To load report as a stream, create a report stream using the **FileStream** class and assign the report stream to the **Stream** property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string filePath = System.Web.Hosting.HostingEnvironment.MapPath("~/App_Data/Product List.rdlc");
    ;
    // Opens the report from application App_Data folder using FileStream
    FileStream reportStream = new FileStream(filePath, FileMode.Open, FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
'
```

In the previous code, the `Product List.rdlc` report is loaded from the `App_Data` folder location.

[View report click](#)

You can get the user selected parameter details when users clicks the `ViewReport` button in the parameter block. The `viewReportClick` event allows you handle the `ViewReport` button click at client side as shown in the following code.

```
'javascript
var viewerStyle = {'height': '700px', 'width': '100%'};
var reportPath = 'GroupingAgg.rdl';
function viewReportClick(event) {
    var reportParams = [];
    reportParams.push({ name: 'ReportParameter1', labels: ['SO50756'], values: ['SO50756'] });
    event.model.parameters = reportParams;
}
function App() {
    return (
        <div style={viewerStyle}>
            <BoldReportViewerComponent
                id="reportviewer-container"
                reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
                processingMode = {"Local"}
                reportPath = {reportPath}
                viewReportClick = {viewReportClick}>
    
```

Render subreport

View report click

```
</BoldReportViewerComponent>
</div>
);
}

export default App;
```

The model property in the event argument has the details of current processing report model.

Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

Add the subreport and main reports to your application `App_Data` folder. In this tutorial, the already created reports are used. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section.

Download the `SideBySideMainReport.rdl` and `SideBySideSubReport.rdl` reports from [here](#). You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#). The reports used from the installed location requires the `NorthwindIO_Reports.sdf` database to run, so add it to your application.

Set the `reportPath` and `reportServiceUrl` properties of the Report Viewer as shown in following code snippet.

```
'javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container">
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer' }
            reportPath = { 'SideBySideMainReport.rdl' }

        </BoldReportViewerComponent>
    </div>);
}
```

Build and run the application.

Change subreport path

To change the subreport file path, set the `ReportPath` property of `SubReportModel` in the `OnInitReportOptions` method.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.ReportPath =
            System.Web.Hosting.HostingEnvironment.MapPath(@"~/AppData/SubReportDetail.rdl");
    }
}
```

Set subreport parameter

You can change the parameter default values of a subreport in the `OnReportLoaded` method of the Web API Controller as given in the following code snippet.

'csharp

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.Parameters = new BoldReports.Web.ReportParameterInfoCollection();
        reportOption.SubReportModel.Parameters.Add(new BoldReports.Web.ReportParameterInfo()
        {
            Name = "SalesPersonID",
            Values = new List<string>() { "2" }
        });
    }
}
```

Modify subreport data source connection string

You can change the credential and connection information of the data sources used in the subreport using the `SubReportModel` in the `OnInitReportOptions` method.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

Render subreport

Set subreport data source

```
{  
if (reportOption.SubReportModel != null)  
{  
    reportOption.SubReportModel.DataSourceCredentials = new  
    List<BoldReports.Web.DataSourceCredentials>();  
  
    reportOption.SubReportModel.DataSourceCredentials.Add(new  
    BoldReports.Web.DataSourceCredentials("NorthWind", "Data  
    Source=dataplatformdemodata.syncfusion.com;Initial Catalog=Northwind;user id=demoreadonly@data-  
    platform-demo;password=N@c)=Y8s*1&dh"));  
}  
}  
`
```

Set subreport data source

You can bind local business object data source collection only for RDLC reports. To specify data source of a RDLC subreport, set the `ReportDataSource` property in the `OnReportLoaded` method.

The RDL report has the connection information in report definition itself, so no need to bind the data source.

Add the RDLC subreport and main reports to your application `App_Data` folder. You can download it from [here](#).

Set the `reportPath` and `reportServiceUrl` properties of the Report Viewer as shown in following code snippet.

```
`javascript  
function App() {  
    return (<div id="viewer" style={viewerStyle}>  
        <BoldReportViewerComponent id="reportviewer-container"  
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer'}  
            reportPath = { 'SideBySideMainReport.rdl' }  
  
        </BoldReportViewerComponent>  
    </div>);  
}  
`
```

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

Render subreport

Set subreport data source

```
`csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList<ProductList> GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
            Category = "Non-Veg",
            Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
            ProductImage = ""
        };
        datas.Add(data);
    }
}
```

Render subreport

Load subreport stream

```
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};

datas.Add(data);
return datas;
}
```

}

`

Bind the business object data values collection to subreport by adding a new item to the **DataSources** as shown in the following code snippet.

```
`csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //Assigning the data source for 'Product List.rdlc'
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
            "list", Value = ProductList.GetData() });
    }
}
```

`

The data source name is case sensitive, and it should be same as in the report definition.

[Load subreport stream](#)

To load subreport as stream, set the **Stream** property in the **OnInitReportOptions** method.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

```
{  
if (reportOption.SubReportModel != null)  
{  
// Opens the report from application App_Data folder using FileStream and loads the sub report stream.  
FileStream reportStream = new  
FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/App_Data/Product List.rdlc"),  
FileMode.Open, FileAccess.Read);  
reportOption.SubReportModel.Stream = reportStream;  
}  
}  
  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
//Assigning the data source for 'Product List.rdlc'  
if (reportOption.SubReportModel != null)  
{  
reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();  
reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =  
"list", Value = ProductList.GetData() });  
}  
}  
}`
```

Report parameters

Provides property options to pass or set report parameters default values at run time using the **parameters** property. You can set the report parameters while creating the Report Viewer control in a script or in the Web API Controller.

In this tutorial, the **Sales Order Detail.rdl** report is used, and it can be downloaded from [here](#).

Set parameter at client side

The **parameters** property takes the JSON array value input with parameter details.

Set the default value data to the **values** property and name of the report parameter to the **name** property.

The parameter name is case sensitive, and it should be same as available in the report definition.

The following code example illustrates how to set a report parameter in the script.

```
'javascript
```

```
var parameters = [{
```

Report parameters

Set parameters in Web API Controller

```
name: 'SalesOrderNumber',
labels: ['SO50751'],
values: [SO50751],
nullable: false
});
function App() {
return (<div id="viewer" style={viewerStyle}>
<BoldReportViewerComponent id="reportviewer-container"
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = { 'Sales Order Detail.rdl' }
parameters = {parameters}

</BoldReportViewerComponent>
</div>);
}
`
```

Build and run the application.

[Set parameters in Web API Controller](#)

To set parameter default value in the Web API Controller, use the following code in the **OnReportLoaded** method.

```
`csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    List<BoldReports.Web.ReportParameter> userParameters = new
    List<BoldReports.Web.ReportParameter>();
    userParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = "SalesOrderNumber",
        Values = new List<string>() { "SO50756" }
    });
    reportOption.ReportModel.Parameters = userParameters;
}
`
```

Get report parameter

The **ReportHelper** class provides methods that help you to get the report parameters used in the report. The following helper methods are used to get parameter with or without values.

Methods | Description

GetParameters | Returns the parameters used in the current report without the processed values.

GetParametersWithValues | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

'csharp

```
public class ReportsApiController : ApiController, IReportController
{
    Dictionary<string, object> jsonArray = null;

    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        jsonArray = jsonResult;
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    .....

    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        var reportParameters = ReportHelper.GetParameters(jsonArray, this);

        List<BoldReports.Web.ReportParameter> setParameters = new
        List<BoldReports.Web.ReportParameter>();

        if (reportParameters != null)
        {
            foreach (var rptParameter in reportParameters)
            {
                setParameters.Add(new BoldReports.Web.ReportParameter()
                {
                    Name = rptParameter.Name,
                    Values = new List<string>() { "SO50756" }
                });
            }
        }

        reportOption.ReportModel.Parameters = setParameters;
    }
}
```

```
}
```

```
}
```

```
}
```

```
,
```

Report interaction events

You can handle the report interaction events with reports using the following events.

ReportLoaded

ReportError

ShowError

Drill through

Hyperlink

Report loaded

The **reportLoaded** event occurs after the report is loaded and it is ready to start the processing. You can handle the event and specify the data source and parameters at client side. The following sample code loads a report by assigning the report data source input in the **reportLoaded** event.

In this tutorial, the **IndicatorReport.rdlc** report is used. You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

```
'javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'IndicatorReport.rdlc' }
            processingMode = { 'Local' }
            reportLoaded = {reportLoaded}

        </BoldReportViewerComponent>
    </div>);
}

function reportLoaded(event) {
    var desc2013 = [
```

```
{  
No: 1, Name: "Carlos Slim", NetWorth: 73.0, Age: 73, CitizenShip: "Mexico", Source: "Telmex,America  
Movil, Grupo Carso", RankingStatus: 50, ProfitStatus: 75  
},  
{  
No: 2, Name: "Bill Gates", NetWorth: 67.0, Age: 57, CitizenShip: "United States", Source: "Microsoft",  
RankingStatus: 50, ProfitStatus: 75  
},  
{  
No: 3, Name: "Amancio Ortega", NetWorth: 57.0, Age: 57, CitizenShip: "Spain", Source: "Inditex Group",  
RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 4, Name: "Warren Buffett", NetWorth: 53.0, Age: 82, CitizenShip: "United States", Source:  
"Berkshire Hathaway", RankingStatus: 25, ProfitStatus: 75  
},  
{  
No: 5, Name: "Larry Ellison", NetWorth: 43.0, Age: 68, CitizenShip: "United States", Source: "Oracle  
Corporation", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 6, Name: "Charles Koch", NetWorth: 34.0, Age: 77, CitizenShip: "United States", Source: "Koch  
Industries", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 7, Name: "David Koch", NetWorth: 34.0, Age: 72, CitizenShip: "United States", Source: "Koch  
Industries", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 8, Name: "Li Ka-shing", NetWorth: 32.0, Age: 84, CitizenShip: "Hong Kong/ Canada", Source:  
"Cheung Kong Holdings", RankingStatus: 75, ProfitStatus: 75  
},  
{  
No: 9, Name: "Liliane Bettencourt", NetWorth: 30.0, Age: 90, CitizenShip: "France", Source: "L'Oreal",  
RankingStatus: 75, ProfitStatus: 75  
},
```

```
{  
No: 10, Name: "Bernard Arnault", NetWorth: 29.0, Age: 63, CitizenShip: "France", Source: "LVMH Moet  
Hennessy Louis Vuitton", RankingStatus: 25, ProfitStatus: 25  
}];  
var desc2012 = [  
{  
No: 1, Name: "Carlos Slim", NetWorth: 69.0, Age: 72, CitizenShip: "Mexico", Source: "Telmex,America  
Movil, Grupo Carso", RankingStatus: 50, ProfitStatus: 25  
,  
{  
No: 2, Name: "Bill Gates", NetWorth: 61.0, Age: 56, CitizenShip: "United States", Source: "Microsoft",  
RankingStatus: 50, ProfitStatus: 75  
,  
{  
No: 3, Name: "Warren Buffett", NetWorth: 44.0, Age: 81, CitizenShip: "United States", Source:  
"Berkshire Hathaway", RankingStatus: 50, ProfitStatus: 25  
,  
{  
No: 4, Name: "Bernard Arnault", NetWorth: 41.0, Age: 63, CitizenShip: "France", Source: "LVMH Moet  
Hennessy Louis Vuitton", RankingStatus: 50, ProfitStatus: 75  
,  
{  
No: 5, Name: "Amancio Ortega", NetWorth: 37.5, Age: 75, CitizenShip: "Spain", Source: "Inditex Group",  
RankingStatus: 75, ProfitStatus: 75  
,  
{  
No: 6, Name: "Larry Ellison", NetWorth: 36.0, Age: 67, CitizenShip: "United States", Source: "Oracle  
Corporation", RankingStatus: 25, ProfitStatus: 75  
,  
{  
No: 7, Name: "Eike Batista", NetWorth: 30.0, Age: 55, CitizenShip: "Brazil", Source: "EBX Group",  
RankingStatus: 75, ProfitStatus: 75  
,  
{  
No: 8, Name: "Stefan Persson", NetWorth: 26.5, Age: 64, CitizenShip: "Sweden", Source: "H&M",  
RankingStatus: 75, ProfitStatus: 75
```

```
},  
{  
    No: 9, Name: "Li Ka-shing", NetWorth: 25.0, Age: 83, CitizenShip: "Hong Kong/ Canada", Source:  
    "Cheung Kong Holdings", RankingStatus: 75, ProfitStatus: 75  
},  
{  
    No: 10, Name: "Karl Albrecht", NetWorth: 25.4, Age: 92, CitizenShip: "Germany", Source: "Aldi",  
    RankingStatus: 75, ProfitStatus: 25  
}];  
  
var description = [  
{  
    Status: 25, Description: "Has not changed from the previous ranking."  
},  
{  
    Status: 50, Description: "Has increased from the previous ranking."  
},  
{  
    Status: 75, Description: "Has decreased from the previous ranking."  
}];  
  
event.model.dataSources = [{  
    value: ej.DataManager(desc2013).executeLocal(ej.Query()),  
    name: "DataSet1"  
}, {  
    value: ej.DataManager(desc2012).executeLocal(ej.Query()),  
    name: "DataSet2"  
}, {  
    value: ej.DataManager(description).executeLocal(ej.Query()),  
    name: "DataSet3"  
}];  
}  
,
```

Report error

When an error occurs in the report processing, it raises the `reportError` event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```
`javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'Sales Order Detail.rdl' }
            reportError = {onReportError}

        </BoldReportViewerComponent>
    </div>);
}

function onReportError(event) {
    alert(event errmsg);
    event.cancel = true;
}
`
```

To suppress the default error dialog, set the cancel argument to true.

[Show error](#)

The `showError` event is invoked whenever users click a report item that contains an error in processing. It provides detailed information about the cause of the error. You can hide the default dialog and show your customized dialog.

```
`javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'Sales Order Detail.rdl' }
            reportError = {onShowError}

        </BoldReportViewerComponent>
    </div>);
}

function onShowError(event) {
    alert("Error code : " + event.errorCode + "\n" +
`
```

```
"Error Detail : " + event.errorDetail + "\n" +
"Error Message : " + event.errorMessage);
event.cancel = true;
}
`
```

Drill through

When a drill through item is selected in a report, it invokes the `drillThrough` event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

```
`javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'SalesPersonDetails.rdl' }
            drillThrough = {onDrillThrough}

        </BoldReportViewerComponent>
    </div>);
}

function onDrillThrough(event) {
    event.actionInfo.ReportName = "PersonalDetails";
    event.actionInfo.Parameters = [
        {
            name: 'SalesOrderNumber',
            labels: ['SO50751'],
            values: [SO50751],
            nullable: false
        }];
}
```

Hyperlink

The `hyperlink` event occurs when users click a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
`javascript
```

Handle post actions

AjaxBeforeLoad

```
function App() {  
    return (<div id="viewer" style={viewerStyle}>  
        <BoldReportViewerComponent id="reportviewer-container"  
            reportServiceUrl = { 'https://demos.boldreports.com/services/api/ReportViewer'}  
            reportPath = { 'Customer Support Analysis (Random data).rdl' }  
            hyperLink = {onHyperLink}  
  
        </BoldReportViewerComponent>  
    </div>);  
}  
  
function onHyperLink(event) {  
    event.cancel = true;  
    //You can modify the URL here  
    window.open(event.actionInfo.Hyperlink);  
}  
`
```

Handle post actions

Report processing actions are sent in an Ajax request to exchange data with the Web API service. You can handle post actions event to customize the Ajax requests.

AjaxBeforeLoad

AjaxSuccess

AjaxError

AjaxBeforeLoad

This event can be triggered before an Ajax request is sent to the Report Viewer Web API service. It allows you to set additional headers and custom data in the Ajax request. The following code sample demonstrates adding custom authorization header and passing default parameter values to service.

Add custom header in Ajax request

Initialize the `ajaxBeforeLoad` event in the script and add the authorization token to the `headers` property.

`javascript

```
function App() {  
    return (<div id="viewer" style={viewerStyle}>  
        <BoldReportViewerComponent id="reportviewer-container"
```

```
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = { 'Sales Order Detail.rdl' }
ajaxBeforeLoad = {onAjaxRequest}

</BoldReportViewerComponent>
</div>);
}
function onAjaxRequest(event) {
event.headers.push({
Key: 'Authorization', Value: 'demo@123'
});
}
`
```

In this tutorial, the Sales Order Detail.rdl report is used, and it can be downloaded from [here](#).

Get the custom header value from the HttpContext header collection using the key name specified at client side.

```
`csharp
string authenticationHeader;

public object PostReportAction(Dictionary<string, object> jsonResult)
{
if (jsonResult != null)
{
//Get client side custom ajax header and store in local variable
authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
//Perform your custom validation here
if (authenticationHeader == "")
{
return new Exception("Authentication failed!!!");
}
else
{
return ReportHelper.ProcessReport(jsonResult, this);
}
}
```

```
}
```

```
return null;
```

```
}
```

```
'
```

Perform your own action to validate the header values.

Pass custom data in Ajax request

Use the `data` property to set custom data to the server in the Ajax request. In the following code sample, parameter values are passed to the server side.

```
'javascript
```

```
function App() {
```

```
    return (<div id="viewer" style={viewerStyle}>
```

```
        <BoldReportViewerComponent id="reportviewer-container"
```

```
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
```

```
            reportPath = { 'Sales Order Detail.rdl' }
```

```
            ajaxBeforeLoad = {onAjaxRequest}
```

```
</BoldReportViewerComponent>
```

```
</div>);
```

```
}
```

```
function onAjaxRequest(event) {
```

```
    event.headers.push({
```

```
        Key: 'Authorization', Value: 'demo@123'
```

```
    });
```

```
    event.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
```

```
}
```

```
'
```

The custom data values are stored in the `customData` header key, and you can store them in the local property. The following code sample demonstrates storing parameter values and setting those values to the report in the `OnReportLoaded` method.

```
'csharp
```

```
public string DefaultParameter = null;
```

```
string authenticationHeader;
```

```
public object PostReportAction(Dictionary<string, object> jsonResult)
```

```
{
```

Handle post actions

AjaxSuccess

```
if (jsonResult != null)
{
    if (jsonResult.ContainsKey("customData"))
    {
        //Get client side custom data and store in local variable. Here parameter values are sent.
        DefaultParameter = jsonResult["customData"].ToString();
    }

    //Get client side custom ajax header and store in local variable
    authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
    //Perform your custom validation here
    if (authenticationHeader == "")
    {
        return new Exception("Authentication failed!!!");
    }
    else
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }
}

return null;
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    if (DefaultParameter != null)
    {
        //Set client side custom header data
        reportOption.ReportModel.Parameters =
JsonConvert.DeserializeObject<List<BoldReports.Web.ReportParameter>>(DefaultParameter);
    }
}
`
```

AjaxSuccess

To perform custom action or show user defined message, use the `ajaxSuccess` event on the successful Ajax request.

```
`javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'Sales Order Detail.rdl' }
            ajaxSuccess = {onAjaxSuccess}

        </BoldReportViewerComponent>
    </div>);
}

function onAjaxSuccess(event) {
    //Perform your custom success message here
    alert("Ajax request success!!!!");
}

`
```

AjaxError

The `ajaxError` event is called, if an error occurred with the Ajax request. You can display the customized error details in the event method.

```
`javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'Sales Order Detail.rdl' }
            ajaxError = {onAjaxFailure}

        </BoldReportViewerComponent>
    </div>);
}

function onAjaxFailure(event) {
    alert("Status: " + event.status + "\n" +
        "Error: " + event.responseText);
}
```

[Print report](#)

[View report in print mode](#)

You can never have both an error and a success callback with a request.

[Print report](#)

The Report Viewer provides print report option in the toolbar to print a copy of the report. The Page Setup dialog allows you to set the paper size or other page setup properties. To see print margins, click **Print Layout** on the toolbar.

You can set values in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values for the Page Setup dialog is based on the report properties set in the design view.

[View report in print mode](#)

Print margins are displayed in the Print Layout only. To view the report in print mode by default, set the **printMode** property to true.

```
'javascript
var isPrintMode = true;

function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            printMode = {isPrintMode}

        </BoldReportViewerComponent>
    </div>);
}
```

By default, the Report Viewer renders the report in normal layout in which the print margins are not displayed.

[Print in new page](#)

To open the print in a new tab of the current browser, set the **printOption** property to **NewTab**. By default, it shows the print dialog in the same page.

```
'javascript
var printOption = ej.ReportViewer.PrintOptions.NewTab;
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
```

Print report

Set page orientation and paper size

```
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = { 'GroupingAgg.rdl' }
printOption = {printOption}
```

```
</BoldReportViewerComponent>
</div>);
}
`
```

The pop-up blocker should be enabled for the page to open the print view in a new tab.

[Set page orientation and paper size](#)

You can specify the print page paper size and orientation at client-side to change the page setup properties by setting the `pageSettings` property.

```
'javascript
var isPrintMode = true;
var pageSettings = {
    orientation: ej.ReportViewer.Orientation.Portrait,
    paperSize: ej.ReportViewer.PaperSize.A3,
};
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container">
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            printMode = {isPrintMode}
            pageSettings = {pageSettings}

        </BoldReportViewerComponent>
    </div>);
}
`
```

[Set report margin](#)

To set margin values to the report page setup, use the `margins` property and specify the value to top, right, bottom, and left.

```
'javascript
```

Print report

Set page height and width

```
var isPrintMode = true;
var pageSettings = {
  margins: {
    top: 0.5,
    right: 0.25,
    bottom: 0.25,
    left: 0.25
  }
};

function App() {
  return (<div id="viewer" style={viewerStyle}>
<BoldReportViewerComponent id="reportviewer-container"
  reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
  reportPath = { 'GroupingAgg.rdl' }
  printMode = {isPrintMode}
  pageSettings = {pageSettings}

</BoldReportViewerComponent>
</div>);
}

`
```

The values set in the margin property is considered as inches input.

[Set page height and width](#)

To set the height and width values to the report page setup, use the `height` and `width` properties.

```
`javascript
var isPrintMode = true;
var pageSettings = {
  height: 2.69,
  width: 28.27
};

function App() {
  return (<div id="viewer" style={viewerStyle}>
<BoldReportViewerComponent id="reportviewer-container"
```

```
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = { 'GroupingAgg.rdl' }
printMode = {isPrintMode}
pageSettings = {pageSettings}

</BoldReportViewerComponent>
</div>);
}
`
```

The values set in the height and width properties is considered as inches input.

Print report with images

When the report has more images, the browser will send the report stream to the print dialog before the images are completely loaded. To load the print report stream with complete images, set the **EmbedImageData** property to true in **OnInitReportOptions** as shown in the following code.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.EmbedImageData = true;
}
`
```

Replace the following code sample in client-side HTML file.

```
`javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container">
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }

        </BoldReportViewerComponent>
    </div>);
}
`
```

In this tutorial, the **Product Details.rdl** report is used, and it can be downloaded from [here](#).

External styles in report printing

While printing a report, the external styles used in the application overrides the printable page style and prints output with incorrect alignments. To avoid the external script overriding, set the `isStyleLoad` property to false, which will print the page using only the Report Viewer styles.

```
'javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            reportPrint = {onReportPrint}

        </BoldReportViewerComponent>
    </div>);
}

function onReportPrint(event) {
    event.isStyleLoad = false;
}
'
```

Show print progress

The Report Viewer provides events that help you show the progress information, when the printing process takes a long time to complete.

To show print progress, follow these steps:

Set the `printProgressChanged` in Report Viewer initialization.

Implement the function and add code samples to show a custom message based on the print progress status as shown in the following code snippet.

```
'javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            printProgressChanged = {onPrintProgressChanged}
```

```
</BoldReportViewerComponent>
</div>);
}

function onPrintProgressChanged(event) {
if (event.stage === "beginPrint") {
console.log(event.stage);
}
if (event.stage === "printStarted") {
console.log(event.stage);
}
else if (event.stage === "preparation") {
console.log(event.stage);
if (event.preparationStage === "dataPreparation") {
console.log(event.preparationStage);
console.log(event.totalPages);
console.log(event.currentPage);
}
}
event.handled = true;
}
}

`
```

Remove empty spaces in printing

The extra blank page is created when the body of your report is too wide for your page. To make the report appear on a single page, all the content within the report body must fit on the physical page, and the body width should be as the following formula:

Body Width <= Page Width - (Left Margin + Right Margin)

For more details to remove the empty pages in the report while designing, refer to the knowledge base article of [report page sizing](#).

Export report

The Report Viewer provides events and properties to control and customize the report exporting functionality.

Export event handling

You can show the progress information, when the exporting process takes long time to complete using the `exportProgressChanged` event.

Set the `exportProgressChanged` event in Report Viewer initialization.

Implement the function and replace the following code samples to get the log message based on the progress stage.

```
'javascript
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            exportProgressChanged = {onExportProgressChanged}>
        </BoldReportViewerComponent>
    </div>);
}

function onExportProgressChanged(event) {
    if (event.stage === "beginExport") {
        console.log(event.stage);
    }
    else if (event.stage === "exportStarted") {
        console.log(event.stage);
    }
    else if (event.stage === "preparation") {
        console.log(event.stage);
        console.log(event.format);
        console.log(event.preparationStage);
    }
    event.handled = true;
}
`
```

Export data visualization items

To export the reports with data visualization components, it is mandatory to configure the web scripts in Report Viewer Web API controller. If the report definition uses chart, gauge and map report items then configure the scripts in Web API as the following steps,

Open the Report Viewer Web API controller.

Configure the below scripts and styles in **OnInitReportOptions** on Web API controller.

jquery-1.10.2.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js

ej2-base.min.js

ej2-data.min.js

ej2-pdf-export.min.js

ej2-svg-base.min.js

ej2-lineargauge.min.js

ej2-circulargauge.min.js

ej.map.min.js

bold.report-viewer.min.js

You can replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
    }
}'
```

```
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
//Gauge component scripts
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
//Map component script
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
//Report Viewer Script
"https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
};

reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
  "https://code.jquery.com/jquery-1.10.2.min.js"
};

}
```

The data visualization components will not export without the above script configurations.

Export data visualization items in azure environment

Report Viewer uses WebBrowser to export the data visualization items to PDF, Word, Excel file formats. The WebBrowser is not supported in azure environment. To overcome this limitation in azure environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

To download PhantomJS application and deploy it on your machine, you should accept it's license terms on [LICENSE](#) and [Third-Party](#) document.

Download PhantomJS from [here](#) and extract the download file.

Copy the PhantomJS.exe file from the extracted bin folder and paste into wwwroot/PhantomJS folder in your application.

Open the Report Viewer Web API controller.

Set the UsePhantomJS property to true and PhantomJSPath property in OnInitReportOptions method.

```
`csharp
// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    reportOption.ReportModel.ExportResources.UsePhantomJS = true;
    reportOption.ReportModel.ExportResources.PhantomJSPath = basePath + @"\PhantomJS\";
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
        //Gauge component scripts
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
        //Map component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
        //Report Viewer Script
        "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
    };
    reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
    {
        "https://code.jquery.com/jquery-1.10.2.min.js"
    };
}
```

The **Scripts** and **Dependent** scripts must be added to export the items. For more details refer to the [export data visualization items](#) section.

Change Excel and Word export format

You can change the default file format to any other file format using the `excelFormat` and `wordFormat` properties. The following code sample changes the default versions.

```
'javascript
var exportSettings = {
    excelFormat: ej.ReportViewer.ExcelFormats.Excel2013,
    wordFormat: ej.ReportViewer.WordFormats.Word2013
}
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            exportSettings = {exportSettings}

        </BoldReportViewerComponent>
    </div>);
}
`
```

Hide specific export type for report

Show or hide the default export types available in the component using the `exportOptions` property. The following code hides the HTML export type from the default export options.

```
'javascript
var exportSettings = {
    exportOptions: ej.ReportViewer.ExportOptions.All & ~ej.ReportViewer.ExportOptions.HTML
}
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
            reportPath = { 'GroupingAgg.rdl' }
            exportSettings = {exportSettings}

        </BoldReportViewerComponent>
    </div>);
}
`
```

```
</div>);  
}  
`
```

PDF export options

The **PDFOptions** provides properties to manage PDF export behaviors. You should set the properties in the **OnInitReportOptions** method of the Web API service.

Export with complex scripts

To export reports with the complex scripts, set the **ComplexScript** property of **PDFOptions** instance to true.

```
`csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()  
    {  
        EnableComplexScript = true  
    };  
}
```

PDF conformance

You can export the report as a **PDF/A-1b** document by specifying the **PdfConformanceLevel.Pdf_A1B** conformance level in the **PdfConformanceLevel** property.

```
`csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()  
    {  
        PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B  
    };  
}
```

Add custom PDF fonts

You can add custom fonts to the PDF exported document by adding the font streams to **Fonts** collection in **PDFOptions** instance.

To add custom fonts to the PDF exported document, follow these steps:

Add the font .ttf files into your application App_Data folder.

In the Solution Explorer, open the properties of the font file and set the property Copy to Output Directory as Copy always.

Initialize the Font collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
    {
        //Load Missing font stream
        Fonts = new Dictionary<string, System.IO.Stream>
        {
            { "Segoe UI",
                System.IO.File.OpenRead(System.Web.Hosting.HostingEnvironment.MapPath(@"~/AppData/fontsymbols.ttf"))
            }
        };
    }
}
```

If any fonts used in the report definition is not installed or available in the local system, then you should load the font stream. In the above code, font_symbols font stream is loaded to export the Sales Order Detail.rdl report as symbols.

Word export options

The WordOptions provides properties to manage Word document export behaviors.

Word document type

You can save the report to the required document version by setting the FormatType property.

```
'csharp
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
    FormatType = BoldReports.Writer.WordFormatType.Docx
};
```

Word document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells and render the word document elements without nested grid layout by setting the `LayoutOption` to `TopLevel`. The `ParagraphSpacing` is the distance value added between two elements in the document.

`'csharp`

```
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
    LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel,
    ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()
    {
        Bottom = 0.5f,
        Top = 0.5f
    }
};
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, added an empty paragraph between two tables.

Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without password. The following are the types of protection:

`AllowOnlyComments`: Adds or modifies only the comments in the Word document.

`AllowOnlyFormFields`: Modifies the form field values in the Word document.

`AllowOnlyRevisions`: Accepts or rejects the revisions in the Word document.

`AllowOnlyReading`: Allows you to view the content only in the Word document.

`NoProtection`: Accesses or edits the Word document contents as normally.

`'csharp`

```
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
    ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading
};
```

Excel export options

The `ExcelOptions` provides properties to manage Excel document export behaviors.

Excel document type

You can save the report to the required excel version by setting the `ExcelSaveType` property.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013  
};
```

Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the `LayoutOption` to `IgnoreCellMerge`.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge  
};
```

Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the `ExcelSheetProtection` or enabling the `ReadOnlyRecommended` properties.

```
'csharp
```

```
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    ReadOnlyRecommended = true,  
    ExcelSheetProtection = ExcelSheetProtection.DeletingColumns  
};
```

PowerPoint export options

You can save the report to the required PowerPoint version by setting the `FormatType` property.

```
'csharp
```

```
reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()  
{
```

```
FormatType = BoldReports.Writer.PPTSaveType.PowerPoint2013  
};  
`
```

CSV export options

The `CsvOptions` allows you change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
`csharp
```

```
reportOption.ReportModel.CsvOptions = new BoldReports.Writer.CsvOptions()  
{  
    Encoding = System.Text.Encoding.Default,  
    FieldDelimiter = ",",  
    UseFormattedValues = false,  
    Qualifier = "#",  
    RecordDelimiter = "@",  
    SuppressLineBreaks = true,  
    FileExtension = ".txt"  
};  
`
```

HTML export options

You can hide the separator added at the end of each page by setting the `HidePageSeparator` property to true.

```
`csharp
```

```
reportOption.ReportModel.HTMLOptions = new BoldReports.Writer.HTMLOptions()  
{  
    HidePageSeparator = true  
};  
`
```

Password protect exported document

Allows you protect the exported document such as PDF, Microsoft Word, Microsoft Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet demonstrates how to encrypt the exported document with user defined password.

```
`csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{
```

```
//PDF encryption
reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions();
reportOption.ReportModel.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
{
    UserPassword = "password"
};

//Word encryption
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
    EncryptionPassword = "password"
};

//Excel encryption
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
    PasswordToModify = "password",
    PasswordToOpen = "password"
};

//PPT encryption
reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()
{
    EncryptionPassword = "password"
};
}
```

Password protection is not supported for HTML export format.

Toolbar customization

You can hide the component toolbar to show customized user interface or to customize the toolbar icons and element's appearances using the templates and Report Viewer toolbar customization properties.

In this tutorial, the Sales Order Detail.rdl report is used, and it can be downloaded from [here](#). You can add the reports from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

Hide parameter block and toolbar items

To hide toolbar items, set the `toolbarSettings` property. The following code can be used to remove the parameter option from the toolbar and hide the parameter block. Use the following code snippet in `App.js` file.

```
'js
var toolbarSettings = {
    items: ~ej.ReportViewer.ToolbarItems.Parameters
}

function App() {
    return (
        <div style={viewerStyle} className="App">
            <BoldReportViewerComponent
                id="reportviewer-container"
                serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}
                reportPath = {'~/Resources/docs/sales-order-detail.rdl'}
                toolbarSettings = {toolbarSettings}>
            </BoldReportViewerComponent>
        </div>
    );
}
`
```

The following code sample hides the print options from the toolbar items which can be used in the `App.js` file.

```
'js
var toolbarSettings = {
    items: ~ej.ReportViewer.ToolbarItems.Print
}

function App() {
    return (
        <div style={viewerStyle} className="App">
            <BoldReportViewerComponent
                id="reportviewer-container"
                serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}
                reportPath = {'~/Resources/docs/sales-order-detail.rdl'}
```

```
toolbarSettings = {toolbarSettings}>
</BoldReportViewerComponent>
</div>
);
}
`
```

Similarly, you can show or hide all other toolbar options with the help of `toolbarSettings.items` enum.

Enable stop option in toolbar

To enable stop option in toolbar, set the `toolbarSettings.items` property to `ej.ReportViewer.ToolbarItems.All`. The following code can be used to enable stop option in toolbar in `App.js` file.

```
`js
var toolbarSettings = {
items: ej.ReportViewer.ToolbarItems.All
}
function App() {
return (
<div style={viewerStyle} className="App">
<BoldReportViewerComponent
id="reportviewer-container"
serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = {'~/Resources/docs/sales-order-detail.rdl'}
toolbarSettings = {toolbarSettings}>
</BoldReportViewerComponent>
</div>
);
}
`
```

Hide toolbar

To hide the Report Viewer toolbar, set the `showToolbar` property to `false`.

```
`js
var toolbarSettings = {
showToolbar: false
}`
```

```
function App() {  
    return (  
        <div style={viewerStyle} className="App">  
            <BoldReportViewerComponent  
                id="reportviewer-container"  
                serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}  
                reportPath = {'~/Resources/docs/sales-order-detail.rdl'}  
                toolbarSettings = {toolbarSettings}>  
            </BoldReportViewerComponent>  
        </div>  
    );  
}  
`
```

Decide or hide the export option

The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. The following code hides the HTML export type from the default export options which is used in `App.js` file.

```
`js  
var exportsettings = {  
    exportOptions: ej.ReportViewer.ExportOptions.All & ~ej.ReportViewer.ExportOptions.HTML  
};  
  
function App() {  
    return (  
        <div style={viewerStyle} className="App">  
            <BoldReportViewerComponent  
                id="reportviewer-container"  
                serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}  
                reportPath = {'~/Resources/docs/sales-order-detail.rdl'}  
                exportSettings = {exportsettings}>  
            </BoldReportViewerComponent>  
        </div>  
    );  
}  
`
```

Add custom items to the export drop-down

To add custom items to the export drop-down available in the Report Viewer toolbar, use the `customItems` property and provide the JSON array of collection input with the `index`, `cssClass` name, and `value` properties. Register the `exportItemClick` event to handle the custom item actions as given in following code snippet, which can be used in `App.js` file.

```
'js
var exportsettings = {
  customItems: [
    {
      index: 2,
      cssClass: '',
      value: 'Text File'
    },
    {
      index: 4,
      cssClass: '',
      value: 'DOT'
    }
  ];
  //Export click event handler
  function onExportItemClick(event) {
    if (event.value === "Text File") {
      //Implement the code to export report as Text
      alert("Text File export option clicked");
    } else if (event.value === "DOT") {
      //Implement the code to export report as DOT
      alert("DOT export option clicked");
    }
  }
  function App() {
    return (
      <div style={viewerStyle} className="App">
        <BoldReportViewerComponent
          id="reportviewer-container"
          serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}
```

```
reportPath = {'~/Resources/docs/sales-order-detail.rdl'}
```

```
exportSettings = {exportsettings}
```

```
exportItemClick = {onExportItemClick}>
```

```
</BoldReportViewerComponent>
```

```
</div>
```

```
);
```

```
}
```

```
'
```

Add custom toolbar item

You can add custom items to Report Viewer toolbar using the `toolbarSettings` property. You should register the `toolBarItemClick` event to handle the newly added custom items action.

Add custom item to exiting toolbar group

To add a custom item to existing toolbar group use the `customItems` property in `toolbarSettings` and provide the JSON array of collection input with the `groupIndex`, `index`, `itemType`, `cssClass` name, and `tooltip` properties as given in following code snippet, which is used in `App.js` file.

```
'js
```

```
var toolbarSettings = {
```

```
showToolbar: true,
```

```
items: ~ej.ReportViewer.ToolbarItems.Print,
```

```
customItems: [{
```

```
groupIndex: 1,
```

```
index: 1,
```

```
type: 'Default',
```

```
cssClass: "e-icon e-mail e-reportviewer-icon",
```

```
id: 'E-Mail',
```

```
tooltip: {
```

```
header: 'E-Mail',
```

```
content: 'Send rendered report as mail attachment'
```

```
}
```

```
}]
```

```
}
```

```
//Toolbar click event handler
```

```
function onToolBarItemClick(event) {
```

```
if (event.value === "CustomItem") {
```

```
//Implement the code to CustomItem toolbar option
alert("Email item toolbar option Clicked");
}

}

function App() {
return (
<div style={viewerStyle} className="App">
<BoldReportViewerComponent
id="reportviewer-container"
serviceUrl={"https://demos.boldreports.com/services/api/ReportViewer"}
reportPath = {'~/Resources/docs/sales-order-detail.rdl'}
toolbarSettings = {toolbarSettings}
toolBarItemClick = {onToolBarItemClick}>
</BoldReportViewerComponent>
</div>
);
}

`
```

Add new toolbar group

To add a new toolbar group and custom items to it, use the `customGroups` property in the `toolbarSettings` and provide the JSON array of collection input with the `groupIndex` and `items` properties. The `items` should have the `itemType`, `cssClass`, and `tooltip` properties as given in following code snippet, which is used in the `App.js` file.

```
`js
var toolbarSettings = {
showToolbar: true,
items: ~ej.ReportViewer.ToolbarItems.Print,
customGroups: [{{
items: [{{
type: 'Default',
cssClass: "e-icon e-mail e-reportviewer-icon CustomGroup",
id: 'CustomGroup',
tooltip: { header: 'CustomGroup', content: 'toolbargroups' }
}},
```

```
{  
  type: 'Default',  
  cssClass: "e-icon e-mail e-reportviewer-icon subCustomGroup",  
  id: 'subCustomGroup',  
  tooltip: { header: 'subCustomGroup', content: 'subtoolbargroups' }  
},  
groupIndex: 3  
}  
}  
  
//Toolbar click event handler  
function onToolBarItemClick(event) {  
  if (event.value === "CustomGroup") {  
    //Implement the code to CustomGroup toolbar option  
    alert("CustomGroup toolbar option clicked");  
  }  
  if (event.value === "subCustomGroup") {  
    //Implement the code to subCustomGroup toolbar option  
    alert("SubCustomGroup toolbar option clicked");  
  }  
}  
  
function App() {  
  return (  
    <div style={viewerStyle} className="App">  
      <BoldReportViewerComponent  
        id="reportviewer-container"  
        serviceUrl={'https://demos.boldreports.com/services/api/ReportViewer'}  
        reportPath = {'~/Resources/docs/sales-order-detail.rdl'}  
        toolbarSettings = {toolbarSettings}  
        toolBarItemClick = {onToolBarItemClick}>  
      </BoldReportViewerComponent>  
    </div>  
  );  
}
```

Custom actions

This section explains you the steps required to add user defined buttons in Report Viewer toolbar and invoke custom actions.

Send a report as an email attachment

This topic describes steps required to create custom email option and share a report as an email attachment to other users.

Add email button in Report Viewer

Create an email button in the toolbar using the `customItems` property with the values such as `groupIndex`, `index`, `itemType`, `cssClass`, and `tooltip`. The `toolBarItemClick` event triggers when you click the email button.

Access the Report Viewer model and create a JSON array for sending requests to the Web API server. You can use the following codes to create an event with custom action.

```
'javascript
var toolbarSettings = {
    showToolbar: true,
    items: ej.ReportViewer.ToolbarItems.All & ~ej.ReportViewer.ToolbarItems.Print,
    customItems: [{
        groupIndex: 1,
        index: 1,
        type: 'Default',
        cssClass: "e-icon e-mail e-reportviewer-icon",
        id: 'E-Mail',
        tooltip: {
            header: 'E-Mail',
            content: 'Send rendered report as mail attachment'
        }
    }]
}
function App() {
    return (<div id="viewer" style={viewerStyle}>
        <BoldReportViewerComponent id="reportviewer-container"
            reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
```

Custom actions

Send a report as an email attachment

```
reportPath = { 'GroupingAgg.rdl' }
toolbarSettings = {toolbarSettings}
toolBarItemClick = {ontoolBarItemClick}

</BoldReportViewerComponent>
</div>);

}

function ontoolBarItemClick(args) {
if (args.value == "E-Mail") {
var proxy = $('#viewer').data('boldReportViewer'),
var Report = proxy.model.reportPath;
var lastsIndex = Report.lastIndexOf("/");
var reportName = Report.substring(lastsIndex + 1);
var requrl = proxy.model.reportServiceUrl + '/SendEmail';
var _json = {
exportType: "PDF", reportViewerToken: proxy._reportViewerToken, ReportName: reportName
};
$.ajax({
type: "POST",
contentType: "application/json; charset=utf-8",
url: requrl,
data: JSON.stringify(_json),
dataType: "json",
crossDomain: true
})
}
}

`
```

Create custom email action

To create custom email action, follow these steps:

Create a new action method **SendEmail** in the Web API service.

Export the report to the required type using the **ReportHelper.GetReport** method to send a report stream as an attachment.

The following code sample exports the report to stream and send it as an attachment to a specified mail address. In the code, the `SmtpClient` method is used to send the report as an email attachment.

```
'csharp
public object SendEmail(Dictionary<string, object> jsonResult)
{
    string _token = jsonResult["reportViewerToken"].ToString();
    var stream = ReportHelper.GetReport(_token, jsonResult["exportType"].ToString());
    stream.Position = 0;
    if (!ComposeEmail(stream, jsonResult["reportName"].ToString()))
    {
        return "Mail not sent !!!";
    }
    return "Mail Sent !!!";
}

public bool ComposeEmail(Stream stream, string reportName)
{
    try
    {
        MailMessage mail = new MailMessage();
        SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");
        mail.IsBodyHtml = true;
        mail.From = new MailAddress("xx@gmail.com");
        mail.To.Add("xx@gmail.com");
        mail.Subject = "Report Name : " + reportName;
        stream.Position = 0;
        if (stream != null)
        {
            ContentType ct = new ContentType();
            ct.Name = reportName + DateTime.Now.ToString() + ".pdf";
            System.Net.Mail.Attachment attachment = new System.Net.Mail.Attachment(stream, ct);
            mail.Attachments.Add(attachment);
        }
        SmtpServer.Port = 587;
```

```
SmtpServer.Credentials = new System.Net.NetworkCredential("xx@gmail.com", "xx");
SmtpServer.EnableSsl = true;
SmtpServer.Send(mail);
return true;
}
catch (Exception ex)
{
return ex.ToString();
}
return false;
}
`
```

In the above code sample, the report is exported to PDF format and sent to users using the `SmtpClient` method.

Build and run the application.

Responsive layout rendering of React Report Viewer

Report Viewer will adaptively render itself with optimal user interfaces for phone, tablet, or desktop form factors. This helps your application to scale elegantly on all form factors with ease. You can enable responsive layout rendering in Report Viewer by setting `isResponsive` property to true.

```
'javascript
var isResponsive = true;
function App() {
return (<div id="viewer" style={viewerStyle}>
<BoldReportViewerComponent id="reportviewer-container"
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = { 'GroupingAgg.rdl' }
isResponsive = { isResponsive }

</BoldReportViewerComponent>
</div>);
}
`
```

Normal layout

The following output shows the normal layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in normal layout](/static/assets/react/report-viewer/images/responsive-layout/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/react/report-viewer/images/responsive-layout/responsive-layout.png)

Localization of Bold Reports React Report Viewer

Localization of React Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. To render the static text with specific culture, refer to the following corresponding culture script files and set culture name to the `locale` property of the Report Viewer.

`ej.localetexts.fr-FR.min.js`

`ej.culture.fr-FR.min.js`

Run the below command, to install the `@boldreports/global` package.

```
`javascript
```

```
npm install @boldreports/global --save
```

```
'
```

Refer to the `ej.localetexts.fr-FR.min.js` and `ej.culture.fr-FR.min.js` script files from `node_modules` in the `App.js` file.

```
`javascript
```

```
import React from 'react';
import './App.css';
//Report Viewer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.lineargauge.min';
```

```
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
import '@boldreports/global/l10n/ej.localetexts.fr-FR.min.js';
import '@boldreports/global/i18n/ej.culture.fr-FR.min.js';
...
...
`
```

Initialize the `locale` property in the `App.js` file as shown like in below code snippet.

```
'javascript
var locale = 'fr-FR';
function App() {
return (<div id="viewer" style={viewerStyle}>
<BoldReportViewerComponent id="reportviewer-container"
reportServiceUrl = {'https://demos.boldreports.com/services/api/ReportViewer'}
reportPath = { 'GroupingAgg.rdl' }
locale = {locale}

</BoldReportViewerComponent>
</div>);
}
`
```

React Report Viewer Reporting Service

The React Report Viewer requires a Web API service to process the report files. The following topics explains how to create reporting Web API service to preview the reports.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

IReportController

The `IReportController` interface has the declaration of action methods that is defined in the Web API Controller for processing the RDL, RDLC, and SSRS report and handling the request from the Report Viewer control. The `IReportController` has the following action methods declaration.

[Methods | Description](#)

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

`csharp

```
public class ReportsController: ApiController, IReportController
{
    /// <summary>
    /// Action (HttpGet) method for getting resource for report.
    /// </summary>
    /// <param name="key">The unique key to get the required resource.</param>
    /// <param name="resourceType">The type of the requested resource.</param>
    /// <param name="isPrinting">If set to <see langword="true"/>, then the resource is generated for printing.</param>
    /// <returns>The object data.</returns>
    public object GetResource(string key, string resourceType, bool isPrinting)
    {
        //Returns the report resource for the requested key.
        return ReportHelper.GetResource(key, resourceType, isPrinting);
    }
    /// <summary>
    /// Report Initialization method that is triggered when report begin processed.
    /// </summary>
    /// <param name="reportOptions">The ReportViewer options.</param>
    public void OnInitReportOptions(ReportViewerOptions reportOptions)
    {
        //You can update report options here
    }
    /// <summary>
    /// Report loaded method that is triggered when report and sub report begins to be loaded.
    /// </summary>
    /// <param name="reportOptions">The ReportViewer options.</param>
    public void OnReportLoaded(ReportViewerOptions reportOptions)
    {
        //You can update report options here
    }
}
```

```
}

/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary < string, object > jsonData)
{
    //Processes the report request and returns the result.
    return ReportHelper.ProcessReport(jsonData, this);
}
}

`
```

Create ASP.NET Web API service

In this section, you will learn how to create a Web API Service for Report Viewer using the new ASP.NET Empty Web Application template.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![ASP.NET Web Application project template](/static/assets/react/report-viewer/images/report-service/aspnetmvc5-template.png)

Choose **Empty, Web API** and then click **OK**. Now, the Web application project is created with Web API.

![Select Web API and Empty options](/static/assets/react/report-viewer/images/report-service/add-web-api.png)

Configure Report Viewer Web API

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web** NuGet packages, and install them in your Web application.

Package | Purpose

PostReportAction | Action (HttpPost) method for posting the request in report process.

OnInitReportOptions | Report initialization method that occurs when the report is about to be processed.

OnReportLoaded | Report loaded method that occurs when the report and sub report start loading.

GetResource | Action (HttpGet) method to get resource for the report.

ReportHelper

The class **ReportHelper** contains helper methods that help to process a Post or Get request from the Report Viewer control and return the response to the Report Viewer control. It has the following methods:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

Right-click **Controller** folder in your project and select **Add > New Item** from the context menu.

Select **Web API Controller Class** from the listed templates and name it as
ReportViewerController.cs

![Provide controller name](/static/assets/react/report-viewer/images/report-service/add-web-api-controller.png)

Click **Add**.

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
public class ReportViewerController : ApiController, IReportController
{
```

```
// Post action for processing the RDL/RDLC report
public object PostReportAction(Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this);
}

// Get action for getting resources from the report
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]
public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

// Method that will be called when initialize the report options before start processing the report
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    // You can update report options here
}

// Method that will be called when reported is loaded
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    // You can update report options here
}

`
```

Add routing information

To configure routing to include an action name in the URI, open the `WebApiConfig.cs` file and change the `routeTemplate` in the `Register` method as follows,

```
'csharp
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
```

```
// Web API configuration and services
// Web API routes
config.MapHttpAttributeRoutes();
config.Routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{action}/{id}",
    defaults: new { id = RouteParameter.Optional }
);
}
}
`
```

Compile and run the Web API service application.

Enable Cross-Origin Requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **Microsoft.AspNet.WebApi.Cors** NuGet packages, and install them in your Web API application.

Call **EnableCors** in **WebApiConfig** to add CORS services to the **Register** method. Replace the following code to allow any origin requests.

```
`csharp
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Add Enable Cors
        config.EnableCors();

        // Web API configuration and services
        // Web API routes
        config.MapHttpAttributeRoutes();
```

```
config.Routes.MapHttpRoute(  
    name: "DefaultApi",  
    routeTemplate: "api/{controller}/{action}/{id}",  
    defaults: new { id = RouteParameter.Optional }  
)  
}  
}  
`
```

To specify the CORS policy for API controller, add the `[EnableCors]` attribute to the controller class. Specify the policy name.

```
`csharp  
[EnableCors(origins: "", headers: "", methods: "*")]  
public class ReportViewerController : ApiController, IReportController  
{  
    // Post action for processing the RDL/RDLC report  
    public object PostReportAction(Dictionary<string, object> jsonResult)  
    {  
        return ReportHelper.ProcessReport(jsonResult, this);  
    }  
    // Get action for getting resources from the report  
    [System.Web.Http.ActionName("GetResource")]  
    [AcceptVerbs("GET")]  
    public object GetResource(string key, string resourcetype, bool isPrint)  
    {  
        return ReportHelper.GetResource(key, resourcetype, isPrint);  
    }  
    // Method that will be called when initialize the report options before start processing the report  
    public void OnInitReportOptions(ReportViewerOptions reportOption)  
    {  
        // You can update report options here  
    }  
    // Method that will be called when reported is loaded  
    public void OnReportLoaded(ReportViewerOptions reportOption)
```

```
{  
// You can update report options here  
}  
}  
`
```

Create ASP.NET Core Web API Service

To create an ASP.NET Core Web API for Report Viewer using a new ASP.NET Core Web Application template, follow these steps:

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Viewer API creation.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/react/report-viewer/images/report-service/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. Right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages** and then search for **BoldReports.Net.Core** package, and install to the application. The following provides detail of the packages and its usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

Newtonsoft.Json | Serialize and deserialize the data for report viewer. It is a mandatory package for the report viewer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer. It should be referred in project when renders the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0.

Configure Web API

The interface **IReportController** has declaration of action methods that are defined in the Web API Controller for processing the RDL, RDLC, and SSRS reports and for handling request from the Report Viewer control. The IReportController has the following action methods declaration:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportViewerController.cs**

![Adding a new controller to the project](/static/assets/react/report-viewer/images/report-service/add-aspnet-core-api-controller.png)

Click **Add**.

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and then implement its methods.

Create local references for the interfaces given in following table.

Interface | Purpose

IMemoryCache | Report Viewer requires a memory cache to store the information of consecutive client request and have the rendered report viewer information in server.

IHostingEnvironment | IHostingEnvironment used to get the report stream from application **wwwroot\Resources** folder.

You cannot load the application report with path information in ASP.NET Core service. So, you should load the report as stream in **OnInitReportOptions**.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-
    detail.rdl", System.IO.FileMode.Open, System.IO.FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}
'
```

You can replace the template code with the following code.

```
'csharp
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
    // Report viewer requires a memory cache to store the information of consecutive client request and
    // have the rendered report viewer information in server.
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    [HttpPost]
    public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
    {
```

```
return ReportHelper.ProcessReport(jsonArray, this, this._cache);
}

// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-
detail.rdl", System.IO.FileMode.Open, System.IO.FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}

// Method will be called when reported is loaded with internally to start to layout process with
ReportHelper.

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

//Get action for getting resources from the report
[ActionName("GetResource")]
[AcceptVerbs("GET")]
// Method will be called from Report Viewer client to get the image src for Image report item.

public object GetResource(ReportResource resource)
{
    return ReportHelper.GetResource(resource, this, _cache);
}

[HttpPost]
public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, _cache);
}
}
```

The `sales-order-detail.rdl` report can be downloaded from [here](#). Also, you can add the report from Bold Reports installation location. For more information on installed sample location, see [Samples and demos](#).

Run the application and use the API URL in the Report Viewer `reportServiceUrl` property.

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

Enable Cross-Origin requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Call `AddCors` in `Startup.ConfigureServices` to add the CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddMvc();
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
    {
        builder.AllowAnyOrigin()
            .AllowAnyMethod()
            .AllowAnyHeader();
    }));
}
```

To specify the CORS policy for the controller, add the `[EnableCors]` attribute to the controller class. Specify the policy name.

```
'csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
public class ReportViewerController : Controller, IReportController
{
    public IActionResult Index()
    {
        return View();
    }
}
```

....
}
,

Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports React Report Viewer.

[Is it possible to hide report parameters?](#)

[Is it possible to hide the export options in Report Viewer?](#)

Is it possible to hide the parameters in Report Viewer

Yes, it is possible to hide the parameters in Report Viewer. On hiding the parameters, the users will not be able to have the parameter block in the Report Viewer. Refer to this [Hide parameter block and toolbar items](#) section.

Is it possible to hide the export options in Report Viewer

Yes, it is possible to hide the export options in Report Viewer. The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. Refer to this [Decide or Hide the export options](#) section.

Bold reporting tools for React

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

Key features

Data Sources --- Supports connection to major data providers such as

Microsoft SQL Server, Oracle, OLEDB and ODBC for exploring data and design reports with a wide range of data sources.

User friendly Environment --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

Report items --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

Report Parameter --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

Expression --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

Add Web Report Designer to a React application

This section explains the steps required to add web Report Designer to a React application.

Prerequisites

Before getting started with the bold report designer, make sure that your development environment includes the following commands.

[Node JS](#) (version 8.x or 10.x)

[NPM](#) (v3.x.x or higher)

Install the Create React App package

Create React App is a simple way to create single-page React application which provides a build setup with no configuration. To install the Create React App globally in your machine, run the following command in Command Prompt.

```
'typescript  
npm install create-react-app -g  
'
```

To learn more about Create React App package refer [here](#)

Create a new React application

To create a new React application, run the following command in Command Prompt.

```
'typescript  
create-react-app reports  
'
```

The `create-react-app` command adds the `react`, `react-dom`, `react-scripts`, and other dependencies required to your React application.

Install the Create React Class

To configure the Report Designer component, change the directory to your application's root folder.

```
'typescript  
cd reports  
'
```

To install the type definitions for `create-react-class`, run the following command in Command Prompt.

```
'typescript  
npm install create-react-class --save  
'
```

Install the Bold Reports React package

To install the Bold Reports React package, run the following command in Command Prompt.

```
'typescript
```

```
npm install @boldreports/react-reporting-components --save
```

Adding scripts reference

Bold Reports needs `window.jQuery` object to render the React components. Hence create a file named `globals.js` and import `jQuery` in `globals.js` file as like the below code snippet.

```
'typescript
import jquery from 'jquery';
import React from 'react';
import createReactClass from 'create-react-class';
import ReactDOM from 'react-dom';
window.React = React;
window.createReactClass = createReactClass;
windowReactDOM = ReactDOM;
window.$ = window.jQuery = jquery;
'
```

Refer the `globals.js` file in `index.js` file as like below code snippet.

```
'typescript
import './globals'
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import * as serviceWorker from './serviceWorker';
ReactDOM.render(<App />, document.getElementById('root'));
'
```

Adding Report Designer component

The Bold Report Designer script and style files need to be imported, in order to run the Bold web designer. Hence, import the following files in `App.js` file.

```
'typescript
/*eslint-disable */
import React from 'react';
import './App.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
```

```

import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
var designerStyle = {
  'height': '700px',
  'width': '100%'
};
function App() {
  return (
    <div style={designerStyle}>
      <BoldReportDesignerComponent
        id="reportdesigner-container">
      </BoldReportDesignerComponent>
    </div>
  );
}
export default App;
`
```

Open the `public\index.html` and refer the following scripts in `<head>` tag.

```

`html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js"></script>
```

```
'
```

Create a Web API service

The web Report Designer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set a Web API service URL

To set a Web API service, open the `App.js` file and replace the existing code with the below code snippet.

```
'javascript
/eslint-disable /
import React from 'react';
import './App.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
var designerStyle = {
  height: '700px',
  width: '100%'}
```

```
};  
function App() {  
  return (  
    <div style={designerStyle} className="App">  
      <BoldReportDesignerComponent  
        id="reportdesigner-container"  
        serviceUrl={'https://demos.boldreports.com/services/api/ReportDesignerWebApi'}>  
      </BoldReportDesignerComponent>  
    </div>  
  );  
}  
export default App;
```

Run the Application

To run the application, run the following command in command prompt.

```
'typescript  
npm run start  
'
```

The `npm run start` command automatically opens your browser to `http://localhost:3000/[report designer output]/(static/assets/react/report-designer/images/Getting-Started_img1.png)`

Deploying the application in production

To deploy the application in production, run the following command in command prompt which will create a folder named `build` to generate the build files.

```
'typescript  
npm run build  
'
```

Add Web Report Designer with Typescript React application

This section explains the steps required to add web Report Designer to a React Typescript application.

Prerequisites

Before getting started with bold web report designer, make sure your development environment includes the following,

[Node JS \(version 8.x or 10.x\)](#)

[NPM \(v3.x.x or higher\)](#)

Install the Create React App Package

Create React App is a simple way to create single-page React application which provides a build setup with no configuration. To install the Create React App globally in your machine, run the following command in Command Prompt.

```
'typescript  
npm install create-react-app -g  
'
```

To learn more about Create React App package refer [here](#)

Create a new React Typescript Application

To create a new React typescript application, run the below command in Command Prompt.

```
'typescript  
create-react-app reports --template typescript  
'
```

The `create-react-app` command adds the `react`, `react-dom`, `react-scripts` and other dependencies required to your react typescript application.

Install Create React Class

To configure the Report Designer component, change the directory to your application's root folder.

```
'typescript  
cd reports  
'
```

To install the type definitions for `create-react-class` run the following command in Command Prompt.

```
'typescript  
npm install create-react-class --save  
npm install @types/create-react-class --save  
'
```

Install Bold Reports React package

To install the Bold Reports React package run the following command in Command Prompt.

```
'typescript  
npm install @boldreports/react-reporting-components --save  
'
```

Install JQuery package

To install the JQuery package run the following command in Command Prompt.

```
'typescript  
npm install @types/jquery --save  
'
```

Adding Scripts reference

Bold Reports needs `window.jQuery` object to render the React components. Hence create a file named `globals.ts` and import `jQuery` in `globals.ts` file as like the below code snippet.

```
'typescript
import jquery from 'jquery';
import React from 'react';
import createReactClass from 'create-react-class';
import ReactDOM from 'react-dom';
(window as any).React = React;
(window as any).createReactClass = createReactClass;
(window as any).ReactDOM = ReactDOM;
(window as any).$ = (window as any).jQuery = jquery;
'
```

Refer the `globals.ts` file in `index.tsx` file as like below code snippet.

```
'typescript
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
'
```

Adding Report Designer component

The web Report Designer script and style files need to be imported in order for the web Report Designer to run. Hence, import the following files in `App.tsx` file.

```
'typescript
/*eslint-disable */
import React from 'react';
import './App.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
```

```

//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
declare let BoldReportDesignerComponent: any;
var designerStyle = {
  'height': '700px',
  'width': '100%'
};
function App() {
  return (
    <div style={designerStyle}>
      <BoldReportDesignerComponent
        id="reportdesigner-container">
      </BoldReportDesignerComponent>
    </div>
  );
}
export default App;
`
```

Open the `public\index.html` and refer the following scripts in `<head>` tag.

```

`html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js"></script>
```

Create Web API service

The web Report Designer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API service.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set Web API service URL

To set Web API service, open the `app.component.ts` file and add the code snippet as in the **constructor**.

```
'javascript
/ eslint-disable /
import React from 'react';
import './App.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-circulargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej2-lineargauge.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
declare let BoldReportDesignerComponent: any;
var designerStyle = {
  'height': '700px',
  'width': '100%'}
```

```
};  
function App() {  
  return (  
    <div style={designerStyle}>  
      <BoldReportDesignerComponent  
        id="reportdesigner-container"  
        serviceUrl={'https://demos.boldreports.com/services/api/ReportDesignerWebApi'}>  
      </BoldReportDesignerComponent>  
    </div>  
  );  
}  
  
export default App;
```

Run the Application

To run the application, run the following command in command prompt

```
'typescript  
npm run start  
'
```

The `npm run start` command automatically opens your browser to `http://localhost:3000/[report designer output]/(static/assets/react/report-designer/images/TS-Getting-Started_img1.png)`

Deploying the application in production

To deploy the application in production we need to generate the build files. Hence to generate the build files run the below command in command prompt which will create a folder named `build`

```
'typescript  
npm run build  
'
```

Add Web Report Designer to a React Boilerplate application

This section explains the steps required to add web [Bold Report Designer](#) to [React Boilerplate application](#).

Prerequisites

Before getting started with the report designer, make sure that you have the following requirements.

[Node JS \(version 8.x or 10.x\)](#)

[NPM \(v3.x.x or higher\)](#)

To quick start with React Boilerplate application, we have already configured our [Bold Reports with React Boilerplate application](#). Those who are wish to directly getting started with Bold Reports React Boilerplate application execute the below commands.

```
'bash  
git clone https://github.com/boldreports/react-boilerplate.git  
cd react-boilerplate  
npm install  
npm start  
'
```

React Boilerplate Application

Download React Boilerplate application from this [link](#) and extract it.

From the extracted folder, execute the following command to install project dependencies.

```
'bash  
npm install  
'
```

Install the Bold Reports React package

To install the [Bold Reports React](#) package, run the following command in command Prompt.

```
'bash  
npm install @boldreports/react-reporting-components --save-dev  
'
```

Also, install `create-react-class` package, which is required by Bold Reports React package.

```
'bash  
npm install create-react-class --save-dev  
'
```

Adding global scripts reference

Since, Bold Reports requires global `jquery`, `React`, `createReactClass` and `ReactDOM` objects to render the Report components, we need to import and assign those to `window` object in a newly created `app/globals.js` file.

```
'typescript  
import jquery from 'jquery';  
import React from 'react';  
import createReactClass from 'create-react-class';
```

```
import ReactDOM from 'react-dom';
window.React = React;
window.createReactClass = createReactClass;
windowReactDOM = ReactDOM;
window.$ = window.jQuery = jquery;
`
```

Import the `globals.js` file in `app/app.js` file as like below code snippet.

```
`javascript
/
```

```
    app.js *
```

This is the entry file for the application, only setup and boilerplate code.

```
*/
import './globals';
// Needed for redux-saga es6 generator support
import '@babel/polyfill';
// Import all the third party stuff
.....
.....
`
```

Configuring webpack

Since, Bold Reports uses `cur` type files, we need to provide support to load such type of file in webpack `url-loader` plugin by configuring `internals/webpack/webpack.base.babel.js` file.

```
`javascript
.....
.....
test: /\.jpg|png|gif|cur$/,
use: [
{
  loader: 'url-loader',
  options: {
```

```
// Inline files smaller than 10 kB  
limit: 10 * 1024,  
},  
}  
....  
....  
`
```

Adding Report Designer component

The Bold Report Designer script and style files are needs to be imported in order to run the web designer. So, import the following scripts and css in `app/app.js` file. Now, use the tag `BoldReportDesignerComponent` to render our designer in the application.

```
'javascript  
....  
....  
import history from 'utils/history';  
import 'sanitize.css/sanitize.css';  
//Report designer source  
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';  
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';  
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';  
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';  
//Data-Visualization  
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';  
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';  
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';  
//Reports react base  
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';  
....  
....  
const MOUNT_NODE = document.getElementById('app');  
const bounds = { height: '800px', width: '100%' };  
const render = messages => {  
  ReactDOM.render(  
    <div style={bounds}>
```

```

<BoldReportDesignerComponent id="reportdesigner-container" >
</BoldReportDesignerComponent>
</div>,
MOUNT_NODE,
);
};

.....
.....
`
```

Open the `app\index.html` and refer the following scripts in `<head>` tag.

```

`html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
`
```

Create a Web API service

The web Report Designer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set a Web API service URL

Bind an online `serviceUrl` to Bold Report designer component in `app/app.js` file like the below snippet.

```

`javascript
.....
....
import history from 'utils/history';
```

```
import 'sanitize.css/sanitize.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
.....
.....
const MOUNT_NODE = document.getElementById('app');
const bounds = { height: '800px', width: '100%'};
const render = messages => {
  ReactDOM.render(
    <div style={bounds}>
      <BoldReportDesignerComponent
        id="reportdesigner-container"
        serviceUrl={'https://demos.boldreports.com/services/api/ReportDesignerWebApi'}>
      </BoldReportDesignerComponent>
    </div>,
    MOUNT_NODE,
  );
};
.....
.....
`
```

Run the Application

To run the app, execute the following command and browse to <http://localhost:3000> to see the application.

```
'bash
```

```
npm start
```

```
![report designer output](/static/assets/react/report-designer/images/react-boilerplate.png)
```

Deploying the application in production

To deploy the application in production, run the following command in command prompt which will create a folder named `build` to generate the build files.

```
'bash
```

```
npm run build
```

Add Web Report Designer to a React Boilerplate application

This section explains the steps required to add web [Bold Report Designer](#) to [React Boilerplate TypeScript application](#).

Prerequisites

Before getting started with the report designer, make sure that you have the following requirements.

[Node JS \(version 8.x or 10.x\)](#)

[NPM \(v3.x.x or higher\)](#)

To quick start with React Boilerplate application, we have already configured our [Bold Reports with React Boilerplate TypeScript application](#). Those who are wish to directly getting started with Bold Reports React Boilerplate TypeScript application execute the below commands.

```
'bash
```

```
git clone https://github.com/boldreports/react-boilerplate-typescript.git
```

```
cd react-boilerplate
```

```
npm install
```

```
npm start
```

React Boilerplate TypeScript Application

Download React Boilerplate application from this [link](#) and extract it.

From the extracted folder, execute the following command to install project dependencies.

```
'bash
```

```
npm install
```

Install the Bold Reports React package

To install the [Bold Reports React](#) package, run the following command in command Prompt.

```
'bash
npm install @boldreports/react-reporting-components --save-dev
'
```

Also, install [create-react-class](#) package, which is required by Bold Reports React package.

```
'bash
npm install create-react-class --save-dev
'
```

Adding global scripts reference

Since, Bold Reports requires global `jquery`, `React`, `createReactClass` and `ReactDOM` objects to render the Report components, we need to import and assign those to `window` object in a newly created `app/globals.ts` file.

```
'typescript
import jquery from 'jquery';
import React from 'react';
import createReactClass from 'create-react-class';
import ReactDOM from 'react-dom';
(window as any).React = React;
(window as any).createReactClass = createReactClass;
(window as any).ReactDOM = ReactDOM;
(window as any).$ = (window as any).jQuery = jquery;
'
```

Import the `globals.ts` file in `app/app.tsx` file as like below code snippet.

```
'javascript
/
app.tsx *

This is the entry file for the application, only setup and boilerplate
code.

*/
```

```
import './globals';
// Needed for redux-saga es6 generator support
import 'react-app-polyfill/ie11';
import 'react-app-polyfill/stable';
// Import all the third party stuff
....
```

Configuring webpack

Since, Bold Reports uses `cur` type files, we need to provide support to load such type of file in webpack `url-loader` plugin by configuring `internals/webpack/webpack.base.babel.js` file.

```
'javascript
.....
.....
test: /\.jpg|png|gif|cur$/,
use: [
{
loader: 'url-loader',
options: {
// Inline files smaller than 10 kB
limit: 10 * 1024,
},
}
.....
.....
'
```

Also, change the image quality in `image-webpack-loader` from `65-90` to `[0.65, 0.90]`.

```
'javascript
.....
.....
{
loader: 'image-webpack-loader',
```

```

options: {
  mozjpeg: {
    enabled: false,
    // NOTE: mozjpeg is disabled as it causes errors in some Linux environments
    // Try enabling it in your environment by switching the config to:
    // enabled: true,
    // progressive: true,
  },
  gifsicle: {
    interlaced: false,
  },
  optipng: {
    optimizationLevel: 7,
  },
  pngquant: {
    quality: [0.65, 0.90],
    speed: 4,
  },
},
}

.....
.....
`
```

[Adding Report Designer component](#)

The Bold Report Designer script and style files are needs to be imported in order to run the web designer. So, import the following scripts and css in `app/app.tsx` file. Now, use the tag `BoldReportDesignerComponent` to render our designer in the application.

```

`javascript
.....
.....
import history from 'utils/history';
import 'sanitize.css/sanitize.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
```

```

import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
.....
....
declare let BoldReportDesignerComponent: any;
const MOUNT_NODE = document.getElementById('app') as HTMLElement;
const bounds = { height: '800px', width: '100%'};
const ConnectedApp = (props: { messages: any }) => (
<div style={bounds}>
<BoldReportDesignerComponent id="reportdesigner-container" >
</BoldReportDesignerComponent>
</div>
);
const render = (messages: any) => {
ReactDOM.render(<ConnectedApp messages={messages} />, MOUNT_NODE);
};
.....
.....
`
```

Open the `app\index.html` and refer the following scripts in `<head>` tag.

```

`html
<!-- Data-Visualization -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js"></script>
```

```
'
```

Create a Web API service

The web Report Designer requires a Web API service to process the data and file actions. You can skip this step and use our online [Web API services](#) to create, edit, and browse reports or you must create any one of the following Web API services.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

Set a Web API service URL

Bind an online `serviceUrl` to Bold Report designer component in `app/app.tsx` file like the below snippet.

```
'javascript
.....
....
import history from 'utils/history';
import 'sanitize.css/sanitize.css';
//Report designer source
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-designer.min';
import '@boldreports/javascript-reporting-controls/Scripts/bold.report-viewer.min';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reports.all.min.css';
import '@boldreports/javascript-reporting-controls/Content/material/bold.reportdesigner.min.css';
//Data-Visualization
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.bulletgraph.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.chart.min';
import '@boldreports/javascript-reporting-controls/Scripts/data-visualization/ej.map.min';
//Reports react base
import '@boldreports/react-reporting-components/Scripts/bold.reports.react.min';
....
....
declare let BoldReportDesignerComponent: any;
const MOUNT_NODE = document.getElementById('app') as HTMLElement;
```

```
const bounds = { height: '800px', width: '100%'};

const ConnectedApp = (props: { messages: any }) => (
  <div style={bounds}>
    <BoldReportDesignerComponent
      id="reportdesigner-container"
      serviceUrl={'https://demos.boldreports.com/services/api/ReportDesignerWebApi'}>
    </BoldReportDesignerComponent>
  </div>
);

const render = (messages: any) => {
  ReactDOM.render(<ConnectedApp messages={messages} />, MOUNT_NODE);
};

....
```

Run the Application

To run the app, execute the following command and browse to <http://localhost:3000> to see the application.

```
'bash
npm start
`

![report designer output](/static/assets/react/report-designer/images/react-boilerplate.png)
```

Deploying the application in production

To deploy the application in production, run the following command in command prompt which will create a folder named `build` to generate the build files.

```
'bash
npm run build
`
```

Load Bold Reports Report Server reports

You can integrate Report Designer with Bold Reports Report Server to create, edit, browse and publish reports using the Report Server built-in Web API service. You need a React application configured with Bold Reports resources to do the following steps:

If you don't have React application with configured Bold Reports resource, then create the app using the [Getting-started](#) steps.

The Report Designer requires the `serviceAuthorizationToken` for report service authorization. Create a token for the user by using the Bold Reports Report Server RESTful API. Refer the below code snippet to generate the token with `index.js` file.

```
`js
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
const $ = window.$;
$(function () {
  var dataValue = "";
  var apiRequest = {userid : 'guest@boldreports.com',password : 'demo' };
  $.ajax({
    type: "POST",
    url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
    data: apiRequest,
    success: function (data) {
      dataValue = data.Token;
      var token = JSON.parse(dataValue);
      // Report Viewer initialization.
    }
  });
});
`);
`
```

You need the `serviceAuthorizationToken` as mandatory for Report Designer to interact with Bold Reports Report Server Service. So, pass the generated token to `App`, in order to set the `serviceAuthorizationToken` for Report Designer. Use the following code snippet in `index.js` file to send the token to `App`.

```
`js
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
```

```

import './index.css';
import App from './App';
const $ = window.$;
$(function () {
  var dataValue = "";
  var apiRequest = {userid : 'guest@boldreports.com',password : 'demo' };
  $.ajax({
    type: "POST",
    url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
    data: apiRequest,
    success: function (data) {
      dataValue = data.Token;
      var token = JSON.parse(dataValue);
      ReactDOM.render(<App {...token}></App>, document.getElementById('root'));
    }
  });
});
```

```

Open the `App.js` file and add token parameter with `App` function to get the `serviceAuthorizationToken` for an application page and set `reportPath`, `serviceAuthorizationToken`, and `reportServiceUrl` for Report Viewer. Also, you have to provide the `serverurl` with `ajaxBeforeLoad` as like in the following code example.

```

`js
function App(token) {
 return (<div id="viewer" style={designerStyle}>
<BoldReportDesignerComponent id="reportdesigner-container"
 serviceUrl = {'https://on-premise-demo.boldreports.com/reporting/reportservice/api/Designer'}
 ajaxBeforeLoad = {onAjaxRequest}
 serviceAuthorizationToken = {token["tokentype"] + " " + token["accesstoken"]}
</BoldReportDesignerComponent>
</div>);
}

```

```

function onAjaxRequest(args) {
 args.headers.push({
 Key: 'serverurl', Value: 'https://on-premise-demo.boldreports.com/reporting/api/site/site1'
 });
}

```

```

Navigate to the root of the application and run using the following command.

```

`typescript
npm run start
`
```

React Report Designer Reporting Service

The React Report Designer requires a Web API service to process the data and file actions. The following topics explains how to create reporting Web API service to create, edit, and browse reports.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

IReportDesignerController

The interface **IReportDesignerController** has the declaration of action methods that are defined in Web API Controller to retrieve data, save, edit and browse reports from the server. The IReportDesignerController has the following action methods declaration.

| Methods | Description |
|---------------------|------------------------------------------------------------------------------------|
| PostDesignerAction | Action (HttpPost) method for posting the request for designer actions. |
| UploadReportAction | Action (HttpPost) method for posted file actions. |
| GetImage | Action (HttpGet) method for getting resource for report images. |
| SetData | Write the resource into storage location. |
| GetData | Read the resource from storage location. |
| PostReportAction | Action (HttpPost) method for posting the request for report process. |
| GetResource | Action (HttpGet) method for getting resource for report. |
| OnInitReportOptions | Report initialization method that is triggered when report begins to be processed. |
| OnReportLoaded | Report loaded method that is triggered when report and sub report begin loading. |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request from the Report Viewer control and returns the response to the Report Viewer control. It has the following methods.

Methods | Description

GetResource | Returns the report resource for the requested key.

ProcessReport | Processes the report request and returns the result.

'csharp

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using BoldReports.Web;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
namespace ReportDesignerAPIService
{
    [System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
    public class ReportingAPIController : ApiController, IReportDesignerController
    {
        /// <summary>
        /// Action (HttpGet) method for getting resource for report images.
        /// </summary>
        /// <param name="key">The unique key for request identification.</param>
        /// <param name="image">The name of requested image.</param>
```

```
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetImage")]
[AcceptVerbs("GET")]
public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}

/// <summary>
/// Action (HttpPost) method for posting the request for designer actions.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing designer request.</param>
/// <returns>The object data.</returns>
public object PostDesignerAction(Dictionary<string, object> jsonData)
{
    //Processes the designer request and returns the result.
    return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);
}

/// <summary>
/// File upload method that is call while upload a file.
/// </summary>
/// <param name="key">The unique key for report designer</param>
/// <param name="itemId">The unique key to get the required resource.</param>
/// <param name="itemData">Gets the resource data.</param>
/// <param name="errorMessage">Returns the error message, if the write action is failed.</param>
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    //You can implement the file save logic
}

/// <summary>
/// File download method that is call while downloading a file.
/// </summary>
/// <param name="key">The unique key for report designer</param>
/// <param name="itemId">The unique key to get the required resource.</param>
```

```
public ResourceInfo GetData(string key, string itemId)
{
    //You can implement the file download logic
}

/// <summary>
/// Action (HttpPost) method for uploaded file actions.
/// </summary>
public void UploadReportAction()
{
    //Processes the designer file upload request's.
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}

/// <summary>
/// Action (HttpGet) method for getting resource to report.
/// </summary>
/// <param name="key">The unique key to get the required resource.</param>
/// <param name="resourcetype">The type of the requested resource.</param>
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]

public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

/// <summary>
/// Report Initialization method that is triggered when report begin processed.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //You can update report options here
}
```

```
}

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
    //Processes the report request and returns the result.
    return ReportHelper.ProcessReport(jsonData, this as IReportController);
}
}
}
`
```

Create ASP.NET Web API Service

In this section, you will learn how to create a Web API Service for Report Designer using the new ASP.NET Empty Web Application template.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![Creating a new ASP.NET Web Application Project](/static/assets/javascript/report-designer/report-service/web-api-images/aspnet-new-project.png)

Then choose the empty application in template and click OK.

![Choosing a empty template in new Project](/static/assets/javascript/report-designer/report-service/web-api-images/aspnet-empty-application.png)

List of dependency libraries

The Web API service configuration requires reporting server-side assembly references.

Right-click the project/solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages...**

![Open nuget manager](/static/assets/javascript/report-designer/report-service/web-api-images/add-dependencies-nuget.png)

Alternatively, click **Tools** in menu, then select **NuGet Package Manager | Manage NuGet Packages for Solution...**

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web** NuGet packages, and install them in your Web Forms application.

![Install reporting web package](/static/assets/javascript/report-designer/report-service/web-api-images/nuget-reporting-web.png)

BoldReports.Web package will install the following required dependencies for Report Designer into your application. Click OK.

Package | Purpose

BoldReports.Web | Builds the server-side implementations.

The above nuget package will automatically install below dependency packages.

Packages | Purpose

Syncfusion.Pdf.AspNet | Exports the report to a PDF.

Syncfusion.DocIO.AspNet | Exports the report to a Word.

Syncfusion.XlsIO.AspNet | Exports the report to an Excel.

Syncfusion.Presentation.AspNet | Exports the report to an PowerPoint.

Install the **Microsoft.AspNet.WebApi.Cors** from Nuget to enable the Cross-Origin Resource Sharing (CORS) for Web API.

`csharp

Install-Package Microsoft.AspNet.WebApi.Cors

This command installs the latest package and updates all dependencies, including the core Web API libraries. Use the -Version flag to target a specific version. Browser security prevents Report Designer making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Add Routing Information

The following steps guide you to configure the routing to include an action name in the URI.

Right-click the project in the solution explorer and select **Add > New item**.

In the Add New Item window, select **Global Application class** and name it as **Global.asax**, and then click Add.

![Adding Global.asax file](/static/assets/javascript/report-designer/report-service/web-api-images/add-global-application-class.png)

Open the code-behind file **Global.asax.cs** and add the following using statement.

```
'csharp
using System.Web.Http;
'
```

Then add the the following code to the **Application_Start** method to register CORS.

```
'csharp
protected void Application_Start(object sender, EventArgs e)
{
    System.Web.Http.GlobalConfiguration.Configuration.EnableCors();
}
'
```

For more information about CORS, see [Configure CORS for WebAPI](#)

Add the following code to the **Application_Start** method to register the Routing configuration:

```
'csharp
protected void Application_Start(object sender, EventArgs e)
{
    System.Web.Http.GlobalConfiguration.Configuration.EnableCors();
    System.Web.Http.GlobalConfiguration.Configuration.Routes.MapHttpRoute(
        name: "DefaultApi",
        routeTemplate: "api/{controller}/{action}/{id}",
    );
}
```

```
defaults: new { id = RouteParameter.Optional });
}
`
```

For more information about routing tables, see [Routing in ASP.NET Web API](#).

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **Web API Controller** class and name it as **ReportingAPIController**, and then click **Add**.

![Adding a new controller to the project](/static/assets/javascript/report-designer/report-service/web-api-images/add-web-api-controller.png)

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Configure Web API

The **IReportDesignerController** interface contains the required actions and helper methods declaration to process the designer file and data actions. The **ReportDesignerHelper** and **ReportHelper** class contains methods that help to process Post or Get request from the control and return the response.

| Methods | Description |
|---------------------|----------------------------------------------------------------------------------------|
| PostDesignerAction | Action (HttpPost) method for posting the request for designer actions. |
| UploadReportAction | Action (HttpPost) method for posted file actions. |
| SetData | Write the resource into storage location. |
| GetData | Read the resource from storage location. |
| PostReportAction | Action (HttpPost) method for posting the request for report process. |
| GetResource | Action (HttpGet) method for getting resource for report. |
| OnInitReportOptions | Report initialization method that is triggered when the report begins to be processed. |
| OnReportLoaded | Report loaded method that occurs when the report and sub report start loading. |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| Methods | Description |
|-------------|----------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |

| ProcessReport | Processes the report request and returns the result. |

ReportHelper

The class `ReportHelper` contains helper methods that help process Post or Get request for report preview action and returns the response to the Report Designer. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

Open the `ReportingAPIController` and add the following using statement.

```
'csharp
using BoldReports.Web;
using BoldReports.Web.ReportDesigner;
using BoldReports.Web.ReportViewer;
using System.IO;
using System.Reflection;
using System.Web;
'
```

Next, add the `[EnableCors]` attribute to the `ReportingAPIController` class.

```
'csharp
[System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
public class ReportingAPIController : ApiController
{
}
'
```

Inherit the `IReportDesignerController` interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
[System.Web.Http.Cors.EnableCors(origins: "", headers: "", methods: "*")]
public class ReportingAPIController : ApiController, IReportDesignerController
{
    private string GetFilePath(string itemName, string key)
```

```
{  
    string targetFolder = HttpContext.Current.Server.MapPath("~/");  
    targetFolder += "Cache";  
    if (!Directory.Exists(targetFolder))  
    {  
        Directory.CreateDirectory(targetFolder);  
    }  
    if (!Directory.Exists(targetFolder + "\\\" + key))  
    {  
        Directory.CreateDirectory(targetFolder + "\\\" + key);  
    }  
    return targetFolder + "\\\" + key + "\\\" + itemName;  
}  
  
[System.Web.Http.ActionName("GetImage")]  
[AcceptVerbs("GET")]  
public object GetImage(string key, string image)  
{  
    return ReportDesignerHelper.GetImage(key, image, this);  
}  
public object PostDesignerAction(Dictionary<string, object> jsonResult)  
{  
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null);  
}  
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)  
{  
    errorMessage = string.Empty;  
    if (itemData.Data != null)  
    {  
        File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);  
    }  
    else if (itemData.PostedFile != null)  
    {  
        var fileName = itemId;
```

```
if (string.IsNullOrEmpty(itemId))
{
    fileName = Path.GetFileName(itemData.PostedFile.FileName);
}

itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));
}

return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

public void UploadReportAction()
{
    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}

[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]

public object GetResource(string key, string resourcetype, bool isPrint)
{
    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
    };
}
```

```
//Gauge component scripts
resourcesPath + @"\bold-reports\common\ej2-base.min.js",
resourcesPath + @"\bold-reports\common\ej2-data.min.js",
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
//Report viewer Script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
    resourcesPath + @"\jquery-1.7.1.min.js"
};

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

public object PostReportAction(Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this as IReportController);
}
}
```

Compile and run the Web API service application.

Create ASP.NET Core Web API Service

In this section, you will learn how to create a ASP.NET Core Web API for Report Designer using the new ASP.NET Core Web Application template.

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Designer API creation.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

![Creating a new ASP.NET Core Web Application Project](/static/assets/javascript/report-designer/report-service/core-api-images/aspnet-new-project.png)

Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/javascript/report-designer/report-service/core-api-images/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. In the Solution Explore, right-click the Dependencies, select Manage NuGet Packages and then search the package **BoldReports.Net.Core** and install to the application. The following provides detail of the packages and its usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core package**.

Newtonsoft.Json | Serialize and deserialize the data or report for report designer. It is a mandatory package for the report designer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer and designer. It should be referred in project when process the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0 .

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Designer NuGet packages.

Register CORS

Browser security prevents Report Designer from making requests to your API Service when both runs in a different domain. To allow access to your API service from a different domain, you must enable cross-origin requests.

Register `AddCors` in `Startup.ConfigureServices` to add CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
    {
        builder.AllowAnyOrigin()
            .AllowAnyMethod()
            .AllowAnyHeader();
    }));
    services.AddMvc();
}
```

For more information about CORS, see [Configure CORS for API](#)

Add API Service

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as `ReportingAPIController`, and then click **Add**.

![Adding a new controller to the project](/static/assets/javascript/report-designer/report-service/core-api-images/add-core-api-controller.png)

While adding API Controller class, name it with the suffix `Controller` that is mandatory.

Configure Web API

The `IReportDesignerController` interface contains the required actions and helper methods declaration to process the designer file and data actions. The `ReportDesignerHelper` and `ReportHelper` class contains methods that help to process Post or Get request from the control and return the response.

| Methods | Description | |
|---------------------------------------------------------------------------------------------|-------------|--|
| | | |
| PostDesignerAction Action (HttpPost) method for posting the request for designer actions. | | |

| | |
|----------------------------------------------------------------------------------------------------------|--|
| UploadReportAction Action (HttpPost) method for posted file actions. | |
| SetData Write the resource into storage location. | |
| GetData Read the resource from storage location. | |
| GetImage Action (HttpGet) method for getting resource for report images. | |
| PostReportAction Action (HttpPost) method for posting the request for report process. | |
| GetResource Action (HttpGet) method for getting resource for report. | |
| OnInitReportOptions Report initialization method that occurs when the report is about to be processed. | |
| OnReportLoaded Report loaded method that occurs when the report and sub report start loading. | |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the Report Designer control and returns the response to the Report Designer control. It has the following methods.

| | |
|----------------------------------------------------------------------|--|
| Methods Description | |
| ----- ----- | |
| GetResource Returns the report resource for the requested key. | |
| ProcessReport Processes the report request and returns the result. | |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request for report preview action and returns the response to the Report Designer. It has the following methods.

| | |
|----------------------------------------------------------------------|--|
| Methods Description | |
| ----- ----- | |
| GetResource Returns the report resource for the requested key. | |
| ProcessReport Processes the report request and returns the result. | |

Open the **ReportingAPIController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
'
```

Add the **{action}** placeholder in the route template, if not contains with default controller attribute.

```
'csharp
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller
```

```
{  
}  
`
```

Next, add the [EnableCors] attribute to the ReportingAPIController class and specify the policy name which given in Startup.ConfigureServices.

```
`csharp  
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]  
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller,
BoldReports.Web.ReportDesigner.IReportDesignerController
{
}  
`
```

Inherit the IReportDesignerController interface, and implement its methods (replace the following code in newly created Web API controller).

```
`csharp  
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
[Route("api/[controller]/[action]")]
public class ReportingAPIController : Controller,
BoldReports.Web.ReportDesigner.IReportDesignerController
{
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    public ReportingAPIController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }
    private string GetFilePath(string itemName, string key)
    {
        string targetFolder = this._hostingEnvironment.WebRootPath + "\\";
        targetFolder += "Cache";
```

```
if (!System.IO.Directory.Exists(targetFolder))
{
    System.IO.Directory.CreateDirectory(targetFolder);
}

if (!System.IO.Directory.Exists(targetFolder + "\\\" + key))
{
    System.IO.Directory.CreateDirectory(targetFolder + "\\\" + key);
}

return targetFolder + "\\\" + key + "\\\" + itemName;
}

public object GetImage(string key, string image)
{
    return ReportDesignerHelper.GetImage(key, image, this);
}

public object GetResource(ReportResource resource)
{
    return ReportHelper.GetResource(resource, this, _cache);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

[HttpPost]
public object PostDesignerAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null, this._cache);
}

public object PostFormDesignerAction()
{
    return ReportDesignerHelper.ProcessDesigner(null, this, null, this._cache);
}
```

```
public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, this._cache);
}

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        System.IO.File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = System.IO.Path.GetFileName(itemData.PostedFile.FileName);
        }
        using (System.IO.MemoryStream stream = new System.IO.MemoryStream())
        {
            itemData.PostedFile.OpenReadStream().CopyTo(stream);
            byte[] bytes = stream.ToArray();
            var writePath = this.GetFilePath(fileName, key);
            System.IO.File.WriteAllBytes(writePath, bytes);
            stream.Close();
            stream.Dispose();
        }
    }
}
```

```
return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = System.IO.File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

[HttpPost]
public void UploadReportAction()
{
    ReportDesignerHelper.ProcessDesigner(null, this, this.Request.Form.Files[0], this._cache);
}
```

Compile and run the Web API service application.

How to queries for Bold Reports Report Designer

This section helps to get the answer for the frequently asked how to queries in Bold Reports Report Designer.

[Open server report using report path on opening Report Designer](#)

[How to add datasource and dataset for Report Designer from application?](#)

How to open server report using report path at the time of opening the Report Designer

Use the `openReport` method in `index.js` file as shown in below code snippet to open a report using report path from report server.

```
'javascrip
import './globals';
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
const $ = window.$;
```

```
$(function () {
var dataValue = "";
var apiRequest ={userid : 'guest@boldreports.com',password : 'demo' };
$.ajax({
type: "POST",
url: "https://on-premise-demo.boldreports.com/reporting/api/site/site1/get-user-key",
data: apiRequest,
success: function (data) {
dataValue = data.Token;
var token = JSON.parse(dataValue);
ReactDOM.render(<App {...token}></App>, document.getElementById('root'));
var designerObj = $('#reportdesigner-container').data('boldReportDesigner')
designerObj.openReport('/Sample Reports/Sales Report');
}
});
});
});
```

How to add the data source and dataset for Report Designer from application

You can add the data for reports in the Report Designer from application level on initializing the Report Designer. This can be achieved using the `addDataSource` and `addDataSet` functions, by which you can add the data source and dataset for the reports at the time of initialization of Report Designer.

Find the following steps to add the data source and dataset for Report Designer from application.

Create a function and bind it with the `create` API as shown in the following code snippet.

```
`javascript
function controlInitialized() {
...
}

function App() {
return (
<div style={designerStyle} className="App">
<BoldReportDesignerComponent
id="reportdesigner-container"
```

```
serviceUrl={'https://demos.boldreports.com/services/api/ReportDesignerWebApi'}
```

```
create={controlInitialized}>
```

```
</BoldReportDesignerComponent>
```

```
</div>
```

```
);
```

```
}
```

```
,
```

Use the `addDatasource` method to add the data source in Report Designer as shown in the following code snippet,

```
'javascript
```

```
var datasource =
```

```
{
```

```
type:'BoldReports.RDL.DOM.DataSource',
```

```
Name:'DataSource1',
```

```
Transaction:false,
```

```
DataSourceReference:null,
```

```
SecurityType:'DataBase',
```

```
ConnectionProperties:{
```

```
type:'BoldReports.RDL.DOM.ConnectionProperties',
```

```
ConnectionString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
```

```
EmbedCredentials:false,
```

```
DataProvider:'SQL',
```

```
IsDesignState:false,
```

```
IntegratedSecurity:false,
```

```
UserName:"",
```

```
PassWord:"",
```

```
Prompt:'Specify the Username and password for DataSource DataSource1',
```

```
CustomProperties:[]
```

```
}
```

```
};
```

```
function controlInitialized(){
```

```
var designerObj = $("#reportdesigner-container").data("boldReportDesigner");
```

```
designerObj.addDataSource(datasource);
```

```
}
```

Use the `addDataSet` method to add dataset in Report Designer as shown in the following code snippet,

```
'javascript
var dataset =
{
  type:'BoldReports.RDL.DOM.DataSet',
  Name:'DataSet1',
  Fields:[
    { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},
    { type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},
    { type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },
    { type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null}
  ],
  Query: {
    type: "BoldReports.RDL.DOM.Query",
    CommandText: "SELECT
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[Huma
nResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM
[HumanResources].[Department]",
    CommandType: 0,
    DataSourceName: "DataSource1",
    QueryDesignerState: {
      type: "BoldReports.RDL.DOM.QueryDesignerState",
      Expressions: null,
      Filters: null,
      Joins: null,
      StoredProcedure: null,
      Tables: [
        {

```

```
type: "BoldReports.RDL.DOM.Table",
Columns: [
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
IsSelected: true, Name: "DepartmentID"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
IsSelected: true, Name: "Name"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
IsSelected: true, Name: "GroupName"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
IsSelected: true, Name: "ModifiedDate"
}
],
Name: "Department",
Schema: "HumanResources",
SchemaLevels: [
{ Name: "HumanResources", SchemaType: "Schema" },
{ Name: "Tables", SchemaType: "Category" },
{ Name: "Department", SchemaType: "Table" }
]
},
QueryParameters: [],
Timeout: 0
},
CaseSensitivity:0,
Collation:null,
AccentSensitivity:0,
KanatypeSensitivity:0,
WidthSensitivity:0,
```

```

Filters:[],
SharedDataSet:null,
InterpretSubtotalsAsDetails:0,
DataSetInfo:null,
DataSetObject:null
};

function controlInitialized(){
var designerObj = $("#reportdesigner-container").data("boldReportDesigner");
designerObj.addDataSet(dataset);
}
`
```

Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

Breaking Changes

When you upgrade from previous versions of Bold Reports to 2.2.23, there are few breaking changes. The following section explains the breaking changes for Bold Reports version 2.2.23.

Designer resource read and write API

The resource write and read API's in `IReportDesignerController` are modified to simplify the resource write and read actions with Bold Report Designer.

Below are the new API details tabulated against old API's.

| Old API | New API | Description |
|-------------|---------|-----------------------------------------------------|
| UploadFile | SetData | Writes the designer resource into storage location. |
| GetFilePath | GetData | Reads the designer resource from storage location. |
| GetFiles | | |
| GetFile | | |

Parameters

SetData

| Parameter Name | Type | Description |
|----------------|--------|------------------------------------|
| key | string | Bold report designer unique token. |

| | | | |
|--------------|---------------|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| itemId | string | | Unique id of designer resource. |
| itemData | object | | Gets the resource data. |
| | Property Name | Type | |
| | Data | byte array | Gets the resource data as byte array. To write an external resource into storage location, pass the resource data in this property. |
| | PostedFile | HttpPostedFile | Gets the uploaded resource data as HttpPostedFile. The resource data uploaded within report designer will be received in this property. |
| errorMessage | string | | Returns the error message, if the write action is failed. |

GetData

| Parameter Name | Types | Description |
|----------------|--------|------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |

Return Type

| API Name | Return Type | | |
|----------|---------------|------------|----------------------------------------------------------|
| SetData | boolean | | |
| GetData | object | | |
| | Property Name | Type | Description |
| | Data | byte array | Contains the requested resource data as byte array. |
| | errorMessage | string | Returns the error message, if the read action is failed. |

Code snippet

| Old snippet | New snippet |
|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>UploadFilecsharppublic bool UploadFile(System.Web.HttpPostedFile httpPostedFile){ //Implement resource write actions}</pre> | <pre>SetDatacsharppublic bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage){ //Implement resource write actions}</pre> |
| <pre>GetFilescsharppublic List GetFiles(FileType fileType){ //Implement resource read actions}</pre> | |
| <pre>GetFilePathcsharppublic string GetFilePath(string fileName){ //Implement actions to get file path}</pre> | <pre>GetDatacsharppublic ResourceInfo GetData(string key, string itemId){ //Implement resource read actions}</pre> |
| <pre>GetFilecsharppublic FileModel GetFile(string filename, bool isOverride){ //Implement resource read actions}</pre> | |

How to section for React Bold Reporting Tools

This section helps to get the answer for the frequently asked how to queries in Bold Reporting Tools.

[How to resolve the Javascript memory heap out issue that occurs while building the React application?](#)

How to resolve the Javascript memory heap out issue that occurs while building the React application

The memory heap out issue occurs when the heap size is not sufficient to run the application. To resolve this issue, open the package.json file, which can be found in the root folder of React application and use --maxoldspace_size=4096 as like in the below code snippet.

```
'typescript
...
...
"scripts": {
  "start": "react-scripts --maxoldspace_size=4096 start",
  "build": "react-scripts --maxoldspace_size=4096 build",
  "test": "react-scripts test",
  "eject": "react-scripts eject"
}
```

...

Reporting tools for ASP.NET Core

Enterprise-class reporting tools for ASP.NET Core development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your core applications.

How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Reporting tools for ASP.NET Core

Enterprise-class reporting tools for ASP.NET Core development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your core applications.

How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

System Requirements

This topic describes the software and hardware requirements for setting up the development environment of Bold Reports ASP.NET Core.

Supported Operating Systems

Windows 7+, 8+

Windows Server 2008 R2+

Framework

The below tool is required for Bold Reports ASP.NET Core.

.NET Core 2.1 or higher

Hardware Requirements

The following hardware requirements are necessary for setting up the development environment of Bold Reports ASP.NET Core:

1 GHZ or faster, 32 bit or 64 bit processor.

1 GB RAM for 32 bit or 2 GB RAM for 64 bit.

Software Requirements

The following software requirements are necessary for setting up the development environment of Bold Reports ASP.NET Core:

Microsoft Visual Studio 2017 or later

Internet Information Services (IIS) 7.0+

Browser Compatibility

IE 9+

Microsoft Edge

Mozilla Firefox 22+

Chrome 17+

Opera 12+

Safari 5+

See Also

[Licensing procedure for deployment](#)

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Microsoft Excel, CSV, Microsoft Word, and HTML formats. Some of the key features are,

Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.

Easily customize each element of Report Viewer and provide events for report processing customization.

Display ssrs rdl report in Bold Reports ASP.NET Core Report Viewer

Create your first ASP.NET Core reporting Web application to display already created SSRS RDL report in Bold Reports ASP.NET Core Report Viewer without using a Report Server using this step-by-step instructions.

Bold Reports ASP.NET Core Report Viewer works in **ASP .NET Core 2.1**, **ASP.NET Core 2.2**, and **ASP.NET Core 3.x** versions.

Create ASP.NET Core application

Start Visual Studio 2019 and click **Create new project**.

Choose **ASP.NET Core Web Application**, and then click **Next**.

Change the project name, and then click **Create**.

In the dropdown with the **ASP.NET Core version**, choose **ASP.NET Core 2.1**.

Select the **Web Application (Model-View-Controller)** template, then click **Create**.

![Creating a new ASP.NET Core Application Project](/static/assets/aspnet-core/report-viewer/images/getting-started/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency libraries

In the Solution Explorer tab right-click the project or solution , and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **BoldReports.AspNet.Core**, **BoldReports.Net.Core** and **System.Data.SqlClient** packages, and install them in your Core application. The following table provides details about the packages and their usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Exports the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core**, and **Syncfusion.XlsIO.Net.Core** packages.

Syncfusion.Pdf.Net.Core | Exports the report to a PDF.

Syncfusion.DocIO.Net.Core | Exports the report to a Word.

Syncfusion.XlsIO.Net.Core | Exports the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

Newtonsoft.Json | Serializes and deserialize data for the Report Viewer. It is a mandatory package for Report Viewer, and the package version should be 10.0.1 or higher.

Refer scripts and CSS

Directly refer all the required scripts and style sheets from [CDN](#) links.

The following scripts and style sheets are mandatorily required to use the Report Viewer.

bold.reports.all.min.css

jquery-1.10.2.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js - Renders the chart item. Add this script, only if your report contains the chart report item.

ej2-base.min.js, **ej2-data.min.js**, **ej2-pdf-export.min.js**, **ej2-svg-base.min.js**, **ej2-lineargauge.min.js** and **ej2-circulargauge.min.js** - Render the gauge item. Add these scripts only if your report contains the gauge report item.

ej.map.min.js - Renders the map item. Add this script only if your report contains the map report item.

bold.report-viewer.min.js

Open the `\Views\Shared_Layout.cshtml` page.

Replace the following code in your `\Views\Shared_Layout.cshtml` page tag.

```
'html
<link
href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet"
/>

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
```

```
<!--Render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<!--Render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
<!--Render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

To learn more about rendering a report with data visualization report items, refer to the [how to render data visualization report items](#) section.

The Report Viewer scripts and styles can be added into your application by installing the **BoldReports.JavaScript** online nuget package.

Tag helper

It is necessary to define the following tag helper within the `_ViewImports.cshtml` page to initialize the Report Viewer component with the tag helper support.

```
`js
@using BoldReports.TagHelpers
@addTagHelper *, BoldReports.AspNetCore
`
```

Configure Script Manager

Open the `~/Views/Shared/_Layout.cshtml` page and add the reporting Script Manager at the end of `<body>` element as in the following code sample.

```
`html
<body>
<div style="min-height: 600px; width: 100%;">
    @RenderBody()
</div>
```

```
@RenderSection("Scripts", required: false)
<!-- Bold Reports script manager -->
<bold-script-manager></bold-script-manager>
</body>
`
```

Initialize Report Viewer

Open the `Index.cshtml` page.

Remove the existing codes and add the following code.

```
`html
<bold-report-viewer id="viewer"></bold-report-viewer>
`
```

Add already created reports

Create a folder `Resources` into the `wwwroot` folder in your application to store the RDL reports.

Download the `sales-order-detail.rdl` from [here](#). For more sample reports, refer to [samples and demos](#) section.

Extract the compressed file and paste the `sales-order-detail.rdl` to `Resources` folder.

The Report Viewer is only for rendering the reports. Refer to the [create RDL report](#) section to create a report.

Configure Web API

The ASP.NET Core Report Viewer requires a Web API service to process the RDL, RDLC, and SSRS report files.

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as `ReportViewerController.cs`

![Adding a new controller to the project](/static/assets/aspnet-core/report-viewer/images/getting-started/add-aspnet-core-api-controller.png)

Click **Add**.

While adding API Controller class, name it with the suffix `Controller` that is mandatory.

Open the **ReportViewController** and add the following using statement.

```
'csharp
using System.IO;
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and then implement its methods.

It is required for processing the reports and for handling request from the Report Viewer.

Create local variables inside the **ReportViewController** class.

```
'csharp
//Report Viewer requires a memory cache to store the information of consecutive client request and
//have the rendered report viewer information in server
private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
// IHostingEnvironment used to get the report stream from application wwwroot\Resources folder..
private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
'
```

Load the report as stream in **OnInitReportOptions** method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    FileStream reportStream = new FileStream(basePath + @"\Resources\" +
        reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}
'
```

You cannot load the report stored in application with path information in ASP.NET Core service.

Set the **Route** attribute for **ReportViewController**.

```
'csharp
```

```
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
...
}
```

You can replace the template code with the following code.

```
`csharp
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
// Report viewer requires a memory cache to store the information of consecutive client request and
// have the rendered Report Viewer information in server.

private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
// IHostingEnvironment used with sample to get the application data from wwwroot.

private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
// Post action to process the report from server based json parameters and send the result back to the
client.

public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
{
    _cache = memoryCache;
    _hostingEnvironment = hostingEnvironment;
}

// Post action to process the report from server based json parameters and send the result back to the
client.

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
{
//Contains helper methods that help to process a Post or Get request from the Report Viewer control
and return the response to the Report Viewer control

return ReportHelper.ProcessReport(jsonArray, this, this._cache);
}
```

```
// Method will be called to initialize the report information to load the report with ReportHelper for processing.  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    string basePath = _hostingEnvironment.WebRootPath;  
    // Here, we have loaded the sales-order-detail.rdl report from application the folder  
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.  
    FileStream reportStream = new FileStream(basePath + @"\Resources\" +  
        reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);  
    reportOption.ReportModel.Stream = reportStream;  
}  
  
// Method will be called when reported is loaded with internally to start to layout process with ReportHelper.  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
}  
  
//Get action for getting resources from the report  
[ActionName("GetResource")]  
[AcceptVerbs("GET")]  
  
// Method will be called from Report Viewer client to get the image src for Image report item.  
public object GetResource(ReportResource resource)  
{  
    return ReportHelper.GetResource(resource, this, _cache);  
}  
[HttpPost]  
public object PostFormReportAction()  
{  
    return ReportHelper.ProcessReport(null, this, _cache);  
}
```

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

Enable cross-origin requests

Browser security prevents the Report Viewer from making requests to your Web API Service when both server-side and client-side requests run in different domains. To allow access to your Web API service from a different domain, enable the cross-origin requests.

Open `Startup.cs` file.

Call `AddCors` in `Startup.ConfigureServices` to add CORS services to the app's service container as in below code.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddMvc();
    services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
    {
        builder.AllowAnyOrigin()
            .AllowAnyMethod()
            .AllowAnyHeader();
    }));
}
```

To specify the CORS policy for `ReportViewerController` add the `[EnableCors]` attribute.

```
'csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
public class ReportViewerController : Controller, IReportController
{
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    ....
    ....
}
```

Set report path and service URL

To render the reports available in the application, of the Report Viewer. You can replace the following code in your Report Viewer page.

Open the `Index.cshtml` page.

Set the `report-path` and `report-service-url` properties as in below.

```
'html
```

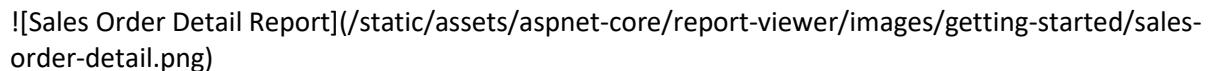
```
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-  
url="/api/ReportViewer"></bold-report-viewer>
```

```
'
```

The report path property is set to the RDL report that is added to the project `Resources` folder.

[Preview the report](#)

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Sales Order Detail Report](/static/assets/aspnet-core/report-viewer/images/getting-started/sales-order-detail.png)

See Also

[Render report with data visualization report items](#)

[Render Report Server reports](#)

[Create RDLC report](#)

[Render RDLC reports](#)

[Preview report in print mode](#)

[Set data source credential for shared data sources](#)

[Change data source connection string](#)

[Create your first app in visual studio 2017](#)

[List of SSRS server versions are supported in Bold Reports](#)

Display SSRS rdl report in ASP.NET Core Razor Pages

Create your first ASP.NET Core Razor Pages application to display already created SSRS RDL report in Bold Reports ASP.NET Core Report Viewer without using a Report Server using this step-by-step instructions.

Bold Reports ASP.NET Core Report Viewer works in `ASP .NET Core 2.1`, `ASP.NET Core 2.2`, and `ASP.NET Core 3.x` versions.

The source code for this ASP.NET Core Razor Pages application is available on [GitHub](#).

Prerequisites

Before getting started with bold web report viewer, make sure your development environment includes the following requirements.

[Visual Studio 2019](#) with `ASP.NET` and `Web Development` workloads.

[.NET Core 3.1](#) Framework.

Create ASP.NET Core application

Start Visual Studio 2019 and click **Create new project**.

Choose **ASP.NET Core Web Application**, and then click **Next**.

Change the project name, and then click **Create**.

In the dropdown with the **ASP.NET Core version**, choose **ASP.NET Core 3.1**.

Select the **Web Application** template, then click **Create**.

![Creating a new ASP.NET Core Razor Pages Application Project](/static/assets/aspnet-core/common/aspnet-core-razor-application-template.png)

List of dependency libraries

In the Solution Explorer tab right-click the project or solution , and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **BoldReports.AspNetCore**, **BoldReports.Net.Core** and **System.Data.SqlClient** packages, and install them in your Core application. The following table provides details about the packages and their usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Exports the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core**, and **Syncfusion.XlsIO.Net.Core** packages.

Syncfusion.Pdf.Net.Core | Exports the report to a PDF.

Syncfusion.DocIO.Net.Core | Exports the report to a Word.

Syncfusion.XlsIO.Net.Core | Exports the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

Newtonsoft.Json | Serializes and deserialize data for the Report Viewer. It is a mandatory package for Report Viewer, and the package version should be 10.0.1 or higher.

Refer scripts and CSS

Directly refer all the required scripts and style sheets from [CDN](#) links.

The following scripts and style sheets are mandatorily required to use the Report Viewer.

bold.reports.all.min.css

jquery-1.10.2.min.js

`bold.reports.common.min.js`

`bold.reports.widgets.min.js`

`ej.chart.min.js` - Renders the chart item. Add this script, only if your report contains the chart report item.

`ej2-base.min.js`, `ej2-data.min.js`, `ej2-pdf-export.min.js`, `ej2-svg-base.min.js`, `ej2-lineargauge.min.js` and `ej2-circulargauge.min.js` - Render the gauge item. Add these scripts only if your report contains the gauge report item.

`ej.map.min.js` - Renders the map item. Add this script only if your report contains the map report item.

`bold.report-viewer.min.js`

Open the `\Pages\Shared_Layout.cshtml` page.

Replace the following code in your `\Pages\Shared_Layout.cshtml` page `<head>` tag.

```
'html
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
<!--Render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<!--Render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
```

```
<!--Render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

To learn more about rendering a report with data visualization report items, refer to the [how to render data visualization report items](#) section.

The Report Viewer scripts and styles can be added into your application by installing the `BoldReports.JavaScript` online nuget package.

Tag helper

It is necessary to define the following tag helper within the `_ViewImports.cshtml` page to initialize the Report Viewer component with the tag helper support.

```
`js
@using BoldReports.TagHelpers
@addTagHelper *, BoldReports.AspNetCore
`
```

Configure Script Manager

Open the `~/Pages/Shared/_Layout.cshtml` page and add the reporting Script Manager at the end of `<body>` element as in the following code sample.

```
`html
<body>
<div>
@RenderBody()
</div>
@RenderSection("Scripts", required: false)
<!-- Bold Reports script manager -->
<bold-script-manager></bold-script-manager>
</body>
`
```

Initialize Report Viewer

Open the `Index.cshtml` page.

Remove the existing codes and add the following code.

```
`html
```

```
<div style="min-height: 600px; width: 100%;">  
<bold-report-viewer id="viewer"></bold-report-viewer>  
</div>  
'
```

Add already created reports

Create a folder **Resources** into the **wwwroot** folder in your application to store the RDL reports.

Download the **sales-order-detail.rdl** from [here](#). For more sample reports, refer to [samples and demos](#) section.

Extract the compressed file and paste the **sales-order-detail.rdl** to **Resources** folder.

The Report Viewer is only for rendering the reports. Refer to the [create RDL report](#) section to create a report.

Configure Web API

The ASP.NET Core Report Viewer requires a Web API service to process the RDL, RDLC, and SSRS report files.

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportViewerController.cs**

![Adding a new controller to the project](/static/assets/aspnet-core/report-viewer/images/getting-started/add-aspnet-core-api-controller.png)

Click **Add**.

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp  
using System.IO;  
using BoldReports.Web.ReportViewer;  
'
```

Inherit the **IReportController** interface, and then implement its methods.

It is required for processing the reports and for handling request from the Report Viewer.

Create local variables inside the **ReportViewerController** class.

```
'csharp
//Report Viewer requires a memory cache to store the information of consecutive client request and
have the rendered report viewer information in server
private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
// IHostingEnvironment used to get the report stream from application wwwroot\Resources folder..
private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
`
```

Load the report as stream in **OnInitReportOptions** method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sales-order-detail.rdl report from application the folder
    // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
    FileStream reportStream = new FileStream(basePath + @"\Resources\" +
        reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}
`
```

You cannot load the report stored in application with path information in ASP.NET Core service.

Set the **Route** attribute for **ReportViewerController**.

```
'csharp
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
    ...
}
```

You can replace the template code with the following code.

```
'csharp
```

```
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
    // Report viewer requires a memory cache to store the information of consecutive client request and
    // have the rendered Report Viewer information in server.
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    [HttpPost]
    public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
    {
        //Contains helper methods that help to process a Post or Get request from the Report Viewer control
        // and return the response to the Report Viewer control
        return ReportHelper.ProcessReport(jsonArray, this, this._cache);
    }
    // Method will be called to initialize the report information to load the report with ReportHelper for
    // processing.
    public void OnInitReportOptions(ReportViewerOptions reportOption)
    {
        string basePath = _hostingEnvironment.WebRootPath;
        // Here, we have loaded the sales-order-detail.rdl report from application the folder
        // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
        FileStream reportStream = new FileStream(basePath + @"\Resources\" +
            reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);
        reportOption.ReportModel.Stream = reportStream;
    }
}
```

```

}

// Method will be called when reported is loaded with internally to start to layout process with
ReportHelper.

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

//Get action for getting resources from the report
[ActionName("GetResource")]
[AcceptVerbs("GET")]

// Method will be called from Report Viewer client to get the image src for Image report item.
public object GetResource(ReportResource resource)
{
    return ReportHelper.GetResource(resource, this, _cache);
}

[HttpPost]
public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, _cache);
}
}
`
```

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

[Web API with ASP.NET Core Web Application](#)

Default ASP.NET Core Razor Pages template (Web Application) does not have MVC configuration to use Web API. So, need to add MVC options with `App` and `Services` using the below steps,

Open `Startup.cs` file.

Call `AddMvcOptions` with `services.AddRazorPages()` as in below code.

```

`csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddRazorPages().AddMvcOptions(option => option.EnableEndpointRouting =
false).AddNewtonsoftJson();
```

```
}
```

```
'
```

Call `UseMvc` with `app` in next of `app.UseRouting()` as in below code.

```
'csharp
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    ....
    ....
    app.UseRouting();
    app.UseMvc();
    ....
    ....
}
```

Set report path and service URL

To render the reports available in the application, of the Report Viewer. You can replace the following code in your Report Viewer page.

Open the `Index.cshtml` page.

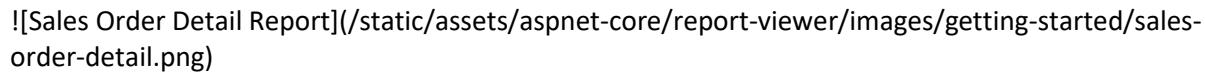
Set the `report-path` and `report-service-url` properties as in below.

```
'html
<div style="min-height: 600px; width: 100%;">
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-
url="/api/ReportViewer"></bold-report-viewer>
</div>
'
```

The report path property is set to the RDL report that is added to the project `Resources` folder.

Preview the report

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Sales Order Detail Report](/static/assets/aspnet-core/report-viewer/images/getting-started/sales-order-detail.png)

See Also

[Render report with data visualization report items](#)

[Render Report Server reports](#)

[Create RDLC report](#)

[Render RDLC reports](#)

[Preview report in print mode](#)

[Set data source credential for shared data sources](#)

[Change data source connection string](#)

[Create your first app in visual studio 2017](#)

[List of SSRS server versions are supported in Bold Reports](#)

Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server.

Pure .Net Core project will support only SSRS 2017 server. The ASP.NET Core project with the combination .NET framework will render report from all SSRS server versions.

To render SSRS Reports, set the `report-server-url`, `report-path`, and `report-service-url` properties as shown in the following code snippet.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-server-url="http://<servername>/reportserver$instanceName" report-
path="/SSRSSamples2/Territory Sales New">
</bold-report-viewer>
'
```

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of `/folder name/report name`.

Network credentials for SSRS

The network credentials are required to load specified SSRS report from the specified SSRS Report Server using the Report Viewer. Specify the `ReportServerCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server credential
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
}
```

Set data source credential to shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS Server. If the report has any data source that uses credentials to connect with the database, then you should specify the **DataSourceCredentials** for each report data source to establish database connection.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-demo",
    "N@c=Y8s*1&dh"));
}
```

Data source credentials should be added to the shared data sources that do not have credentials in the connection strings.

Change data source connection string

You can change the connection string of a report data source before it is loaded in the Report Viewer. The **DataSourceCredentials** class provides the option to set and update the modified connection string as in the following code snippet.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "demoreadonly@data-platform-
    demo", "N@c=Y8s*1&dh", "Data Source=dataplatformdemodata.syncfusion.com;Initial
    Catalog=AdventureWorks"));
}
```

The previous code shows an option to change the connection string only, but the class provides multiple options to change data source information. To learn more about this, refer to the **DataSourceCredentials** class.

Render linked reports

You can render a linked report that points to an existing report, which is published in the SSRS Report Server. You can set the parameter, data source, credential, and other properties like normal SSRS reports using the Report Viewer.

```
'html  
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"  
report-server-url="http://<servername>/reportserver$instanceName" report-  
path="/SSRSSamples/Territory Sales_Link">  
</bold-report-viewer>  
'
```

The Territory Sales_Link is a linked report created for the Territory Sales report that is already available in the SSRS Report Server.

Load SharePoint Server reports

To render the SharePoint server reports, set the report-server-url, report-path, and report-service-url properties as shown in the following code snippet.

```
'html  
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"  
report-server-url="http://<servername>/reportserver$instanceName" report-  
path="http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl">  
</bold-report-viewer>  
'
```

In SharePoint integrated mode, the report-server-url will be same as your site URL. The report-path is relative to the Report Server URL with the file extension.

Forms credential for SharePoint Server

The forms credentials are required to load the SharePoint integrated SSRS report from the specified SharePoint integrated SSRS Report Server using the Report Viewer. Specify the ReportServerFormsCredential property in the Web API Controller OnInitReportOptions method.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    //Add ReportServerFormsCredential for server  
    reportOption.ReportModel.ReportServerFormsCredential = new  
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");  
}
```

Set data source credential to shared data sources

The shared data source credentials can be added to the DataSourceCredentials property to connect with the database.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

Load Bold Report Server reports

Set data source credential to shared data sources

```
{  
//Add ReportServerFormsCredential and data source credentials  
reportOption.ReportModel.ReportServerFormsCredential = new  
BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");  
reportOption.ReportModel.DataSourceCredentials.Add(new  
BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));  
}  
`
```

Data source credentials should be added to shared data sources that do not have credentials in the connection strings.

Load Bold Report Server reports

You can render the Bold Report Server reports in the Report Viewer easily without creating a Web API service. Bold Report Server provides the built-in Web API service that helps you to display the server reports.

The Report Viewer requires the `ServiceAuthorizationToken` to connect and download the items from the Bold Report Server.

To load the Report Server reports, follow these steps:

Create a token for users by using the server RESTful API. The following code is used to generate the token.

```
'csharp  
public partial class ReportViewerController : Controller  
{  
public ActionResult Index()  
{  
ViewBag.ServiceAuthorizationToken = this.GenerateToken( "guest@boldreports.com", "demo");  
return View();  
}  
public IActionResult Error()  
{  
return View();  
}  
public string GenerateToken(string userName, string password)  
{  
using (var client = new HttpClient())
```

Load Bold Report Server reports

Set data source credential to shared data sources

```
{  
client.DefaultRequestHeaders.Accept.Clear();  
  
var content = new FormUrlEncodedContent(new[]  
{  
    new KeyValuePair<string, string>("grant_type", "password"),  
    new KeyValuePair<string, string>("username", userName),  
    new KeyValuePair<string, string>("password", password)  
});  
  
var result = client.PostAsync("https://on-premise-  
demo.boldreports.com/reporting/api/site/site1/token", content).Result;  
string resultContent = result.Content.ReadAsStringAsync().Result;  
var token = JsonConvert.DeserializeObject<Token>(resultContent);  
return token.token_type + " " + token.access_token;  
}  
}  
}  
}  
  
public class Token  
{  
    public string access_token { get; set; }  
    public string token_type { get; set; }  
    public string expires_in { get; set; }  
    public string userName { get; set; }  
    public string serverUrl { get; set; }  
    public string password { get; set; }  
}
```

,

Set the Bold Report Server built-in service URL to the **report-service-url** property.

Assign the created token to **service-authorization-token** and **report-path** of a report deployed on the server. You can use the following complete code in your CSHTML page.

```
`html  
<bold-report-viewer id="viewer" report-service-url="https://on-premise-  
demo.boldreports.com/reporting/reportservice/api/Viewer" report-path="/Sample Reports/Company
```

```
Sales" service-authorization-token="@ViewBag.ServiceAuthorizationToken" ajax-before-load="onAjaxRequest" ></bold-report-viewer>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'serverurl', Value: 'https://on-premise-demo.boldreports.com/reporting/api/site/site1' });
}
</script>
`
```

You can also load the report using GUID instead of report location. Set the GUID of the report in the **report-path** property as **report-path: '91f24bf1-e537-4488-b19f-b37f77481d00'**.

Render RDLC report

The data binding support allows you to view the RDLC reports that exist on the local file system with JSON array and custom business object data collection. The following steps demonstrates how to render a RDLC report with JSON array and custom business object data collection.

Add the RDLC report **Product List.rdlc** from Bold Reports installation location to your application **wwwroot/Resources** folder. For more information, see [Samples and demos](#).

Bind data source in Web API controller

The following steps help you to configure the Web API to render the RDLC report with business object data collection.

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }
    public static IList GetData()
    {
        List<ProductList> datas = new List<ProductList>();
```

```
ProductList data = null;
data = new ProductList()
{
    ProductName = "Baked Chicken and Cheese",
    OrderId = "323B60",
    Price = 55,
    Category = "Non-Veg",
    Ingredients = "grilled chicken, corn and olives.",
    ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
    ProductName = "Chicken Delite",
    OrderId = "323B61",
    Price = 100,
    Category = "Non-Veg",
    Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
    ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};

datas.Add(data);
return datas;
}
```

```
}
```

```
'
```

Set the value of the **ProcessingMode** property to **ProcessingMode.Local** in the RDLC report location.

To load a report as a stream, create a report stream using the **FileStream** class, and assign the report stream to the **Stream** property.

Bind the business object data values collection by adding a new item to the **DataSources** as in the following code snippet.

```
'csharp
```

```
private IMemoryCache _cache;  
private IHostingEnvironment _hostingEnvironment;  
public ReportViewerController(IMemoryCache memoryCache, IHostingEnvironment  
hostingEnvironment)  
{  
    _cache = memoryCache;  
    _hostingEnvironment = hostingEnvironment;  
}  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    string basePath = _hostingEnvironment.WebRootPath;  
    reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;  
    FileStream inputStream = new FileStream(basePath + @"\Resources\Product List.rdlc", FileMode.Open,  
    FileAccess.Read);  
    reportOption.ReportModel.Stream = inputStream;  
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",  
    Value = ProductList.GetData() });  
}
```

```
'
```

Here, the **Name** is case sensitive and it should be same as in the data source name in the report definition.

The **Value** accepts **IList**, **DataSet**, and **DataTable** inputs.

In the previous code, the **Product List.rdlc** report is loaded from the **wwwroot/Resources** folder location.

Bind data source with razor view

The following steps help you to provide the data source for a Report Viewer in razor view using the ViewBag.

Create a class and methods that returns business object data collection. Use the following code in your application.

```
'csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }
    public static IList GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
```

Render RDLC report

Bind data source with razor view

```
Category = "Non-Veg",
Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};
datas.Add(data);
return datas;
}
}

`
```

Create **BoldReports.Models.ReportViewer.ReportDataSource** collection with business object collection as follows and assign to the ViewBag to provide the data for report dataset.

```
`csharp
public ActionResult Index()
{
    BoldReports.Models.ReportViewer.ReportDataSource reportDataSource = new BoldReports.
    Models.ReportViewer.ReportDataSource();
    reportDataSource.Name = "list";
    reportDataSource.Value = ProductList.GetData();
    ViewBag.dataSources = new List<BoldReports.Models.ReportViewer.ReportDataSource> {
        reportDataSource };
    return View();
}
`
```

Pass the ViewBag value to the **dataSources** property of Report Viewer.

```
'html
<bold-report-viewer id="viewer"
report-service-url="/api/ReportViewer"
dataSources="ViewBag.dataSources"
processing-mode="Local" >
</bold-report-viewer>
'
```

[View report click](#)

You can get the user selected parameter details when users click the **ViewReport** button in the parameter block. The **view-report-click** event allows you to handle the **ViewReport** button click at client-side as shown in the following code.

```
'html
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-
url="/api/ReportViewer" view-report-click="onViewReportClick" processing-mode="Remote"></bold-
report-viewer>

<script type="text/javascript">
function onViewReportClick(args) {
args[0] = ({ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'], nullable: false });
}
</script>
'
```

The **model** property in the event argument has the details of current processing report model.

[Render subreport](#)

You can display another report inside the body of main report using the Report Viewer. The following steps help you to customize the subreport properties such as data source, report path, and parameters.

Add the sub report and main reports to the application **wwwroot/Resources** folder. In this tutorial, the already created reports are used. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section.

Download the **SideBySideMainReport.rdl**, **SideBySideSubReport.rdl** reports from [here](#). You can add a report from the Bold Reports installation location. For more information, refer to [Samples and demos](#). The reports used from installed location requires **NorthwindIO_Reports.sdf** database to run, so add the database to your application.

Set the `report-path`, `processing-mode`, and `report-service-url` properties of the Report Viewer as in the following code snippet.

The following code example demonstrates how to load a subreport in the Report Viewer at client side.

`'html`

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" report-path="SideBySideMainReport.rdl" processing-mode="Remote"></bold-report-viewer>
`
```

The following code example demonstrates how to load a subreport in the Report Viewer at server side.

`'csharp`

```
public class ReportViewerController : Controller, IReportController
{
    // Report viewer requires a memory cache to store the information of consecutive client request and
    // have the rendered report viewer information in server.
    private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
        Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }
    // Post action to process the report from server based json parameters and send the result back to the
    // client.
    [HttpPost]
    public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
    {
        return ReportHelper.ProcessReport(jsonArray, this, this._cache);
    }
    // Method will be called to initialize the report information to load the report with ReportHelper for
    // processing.
    public void OnInitReportOptions(ReportViewerOptions reportOption)
```

```
{  
string basePath = _hostingEnvironment.WebRootPath;  
// Here, we have loaded the SideBySideSubReport.rdl report from application the folder  
wwwroot\Resources and loads the sub report stream.  
if (reportOption.SubReportModel != null)  
{  
FileStream SubStream = new FileStream(basePath + @"\Resources\SideBySideSubReport.rdl",  
 FileMode.Open, FileAccess.Read);  
reportOption.SubReportModel.Stream = SubStream;  
}  
else  
{  
FileStream inputStream = new FileStream(basePath + @"\Resources\" +  
 reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);  
reportOption.ReportModel.Stream = inputStream;  
}  
}  
  
// Method will be called when reported is loaded with internally to start to layout process with  
ReportHelper.  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
}  
  
//Get action for getting resources from the report  
[ActionName("GetResource")]  
[AcceptVerbs("GET")]  
// Method will be called from Report Viewer client to get the image src for Image report item.  
public object GetResource(ReportResource resource)  
{  
return ReportHelper.GetResource(resource, this, _cache);  
}  
[HttpPost]  
public object PostFormReportAction()  
{  
return ReportHelper.ProcessReport(null, this, _cache);  
}
```

Render subreport

Change subreport path

```
}
```

```
}
```

```
'
```

Change subreport path

To change the subreport file path, set the Stream property of SubReportModel in the OnInitReportOptions method.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        FileStream SubStream = new FileStream(basePath + @"\Resources\SubReport_Detail.rdl",
        FileMode.Open, FileAccess.Read);
        reportOption.SubReportModel.Stream = SubStream;
    }
}
```

Set subreport parameter

You can change the parameter default values of a subreport in the OnReportLoaded method of the Web API Controller as given in the following code snippet.

```
'csharp
```

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.Parameters = new BoldReports.Web.ReportParameterInfoCollection();
        reportOption.SubReportModel.Parameters.Add(new BoldReports.Web.ReportParameterInfo()
        {
            Name = "SalesPersonID",
            Values = new List<string>() { "2" }
        });
    }
}
```

Modify subreport data source connection string

You can change the credential and connection information of the data sources used in the subreport using the SubReportModel in the `OnInitReportOptions` method.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSourceCredentials = new
List<BoldReports.Web.DataSourceCredentials>();

        BoldReports.Web.DataSourceCredentials dataSourceCredentials = new
        BoldReports.Web.DataSourceCredentials();

        dataSourceCredentials.Name = "NorthWind";

        dataSourceCredentials.ConnectionString = "Data Source=dataplatformdemodata.syncfusion.com;Initial
Catalog=Northwind;user id=demoreadonly@data-platform-demo;password=N@c=Y8s*1&dh";

        reportOption.SubReportModel.DataSourceCredentials.Add(dataSourceCredentials);
    }
}
```

Set subreport data source

You can bind local business object data source collection only for RDLC reports. To specify data source of a RDLC subreport, set the `ReportDataSource` property in the `OnReportLoaded` method.

The RDL report has the connection information in report definition itself, so no need to bind data source.

Add the RDLC sub report and main reports to your application `wwwroot/Resources` folder. You can download it from [here](#).

Set the `report-path`, `processing-mode`, and `report-service-url` properties of the Report Viewer as shown in following code snippet.

'js

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" report-path="Product List
Main.rdlc" processing-mode="Local"></bold-report-viewer>
```

Create a class and methods that returns business object data collection. Use the following code in the application Web API Service.

Render subreport

Set subreport data source

```
`csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList<ProductList> GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
            Price = 100,
            Category = "Non-Veg",
            Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
            ProductImage = ""
        };
        datas.Add(data);
    }
}
```

Render subreport

Load subreport stream

```
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};

datas.Add(data);
return datas;
}
```

}

`

Bind the business object data values collection to the subreport by adding a new item to the **DataSources** as shown in the following code snippet.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //Assigning the data source for 'Product List.rdlc'
    if (reportOption.SubReportModel != null)
    {
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
        "list", Value = ProductList.GetData() });
    }
}
```

`

The data source name is case sensitive, it should be same as in the report definition.

[Load subreport stream](#)

To load subreport as stream, set the **Stream** property in the **OnInitReportOptions** method.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

```
{  
    string basePath = _hostingEnvironment.WebRootPath;  
    if (reportOption.SubReportModel != null)  
    {  
        // Here, we have loaded the Product List.rdlc report from application the folder wwwroot\Resources  
        // and loads the sub report stream.  
        FileStream SubStream = new FileStream(basePath + @"\Resources\Product List.rdlc", FileMode.Open,  
        FileAccess.Read);  
        reportOption.SubReportModel.Stream = SubStream;  
    }  
    else  
    {  
        FileStream inputStream = new FileStream(basePath + @"\Resources\" +  
        reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);  
        reportOption.ReportModel.Stream = inputStream;  
    }  
}  
  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    //Assigning the data source for 'Product List.rdlc'  
    if (reportOption.SubReportModel != null)  
    {  
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();  
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =  
            "list", Value = ProductList.GetData() });  
    }  
}
```

Report parameters

Provides property options to pass or set report parameters default values at run time using the **parameters** property. You can set the report parameters while creating the Report Viewer control in a script or in the Web API Controller.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#).

Set a parameter with razor view

Report Viewer **parameter** property does not have support to create parameter collection with razor view. So, we have to create and assign the parameter with help of **ViewBag** as follows.

```
'csharp
public ActionResult Index()
{
    List<BoldReports.Models.ReportViewer.ReportParameter> parameters = new
    List<BoldReports.Models.ReportViewer.ReportParameter>();
    parameters.Add(new BoldReports.Models.ReportViewer.ReportParameter()
    {
        Name = "SalesOrderNumber",
        Values = new List<string>() { "SO50756" }
    });
    ViewBag.parameters = parameters;
    return View();
}

'
`html
<bold-report-viewer id="reportviewer1" report-service-url="..../ReportApi" processing-mode="Remote"
parameters="ViewBag.parameters" />
`
```

Set parameters in Web API Controller

To set parameter default value in Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    List<BoldReports.Web.ReportParameter> userParameters = new
    List<BoldReports.Web.ReportParameter>();
    userParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = "SalesOrderNumber",
        Values = new List<string>() { "SO50756" }
    });
    reportOption.ReportModel.Parameters = userParameters;
```

```
}
```

[Get report parameter](#)

The `ReportHelper` class provides methods to get the report parameters used in the report. The following helper methods used to get parameter with or without values.

[Methods](#) | [Description](#)

`GetParameters` | Returns the parameters used in the current report without the processed values.

`GetParametersWithValues` | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

```
'csharp
```

```
public class ReportViewerController : ApiController, IReportController
{
    Dictionary<string, object> jsonArray = null;
    private IMemoryCache _cache;
    private IHostingEnvironment _hostingEnvironment;
    public ReportViewerController(IMemoryCache memoryCache, IHostingEnvironment hostingEnvironment)
    {
        _cache = memoryCache;
        _hostingEnvironment = hostingEnvironment;
    }
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        jsonArray = jsonResult;
        return ReportHelper.ProcessReport(jsonResult, this);
    }
    ....
    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        var reportParameters = ReportHelper.GetParameters(jsonArray, this, _cache);
        List<BoldReports.Web.ReportParameter> setParameters = new
        List<BoldReports.Web.ReportParameter>();
        if (reportParameters != null)
        {
```

Report interaction events

Report loaded

```
foreach (var rptParameter in reportParameters)
{
    setParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = rptParameter.Name,
        Values = new List<string>() { "SO50756" }
    });
}

reportOption.ReportModel.Parameters = setParameters;
}
```

Report interaction events

You can handle the report interaction events with reports using the following events.

ReportLoaded

ReportError

ShowError

Drill through

Hyperlink

Report loaded

The **report-loaded** event occurs after the report is loaded and it is ready to start the processing. You can handle the event and specify the data source and parameters at client side. The following sample code loads a report by assigning the report data source input in the **reportLoaded** event.

In this tutorial, the **Product List.rdlc** report is used. You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Local"
report-path="Product List.rdlc" report-loaded="onReportLoaded">
</bold-report-viewer>
<script type="text/javascript">
```

```

function onReportLoaded(args) {
var dataSource = [
{
ProductName: "Baked Chicken and Cheese", OrderId: "323B60", Price: 55, Category: "Non-Veg",
Ingredients: "Grilled chicken, Corn and Olives.", ProductImage: ""

},
{
ProductName: "Chicken Delite", OrderId: "323B61", Price: 100, Category: "Non-Veg", Ingredients:
"Cheese, Chicken chunks, Onions & Pineapple chunks.", ProductImage: ""

},
{
ProductName: "Chicken Tikka", OrderId: "323B62", Price: 64, Category: "Non-Veg", Ingredients: "Onions,
Grilled chicken, Chicken salami & Tomatoes.", ProductImage: ""

}];

var reportObj = $('#viewer').data("boldReportViewer");
reportObj.model.dataSources = [{

value: ej.DataManager(dataSource).executeLocal(ej.Query()),

name: "list"

}];

}
</script>
`
```

Report error

When an error occurs in the report processing, it raises the `report-error` event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```

`html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Local"
report-path="Product List.rdlc" report-error="onReportError">

</bold-report-viewer>
<script type="text/javascript">
function onReportError(args) {
alert(args.errmsg);
args.cancel = true;
}
</script>
```

To suppress the default error dialog, set the cancel argument to true.

Show error

The **show-error** event is invoked whenever users click a report item that contains an error in processing. It provides detailed information about the cause of error. You can hide the default dialog and show your customized dialog.

'html

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Local"
report-path="Product List.rdlc" show-error="onShowError">
</bold-report-viewer>
<script type="text/javascript">
function onShowError(args) {
    alert("Error code : " + args.errorCode + "\n" +
    "Error Detail : " + args.errorDetail + "\n" +
    "Error Message : " + args.errorMessage);
    args.cancel = true;
}
</script>
```

Drill through

When a drill through item is selected in a report, it invokes the **drill-through** event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

'html

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-path="SalesPersonDetails.rdl" drill-through="onDrillThrough">
</bold-report-viewer>
<script type="text/javascript">
function onDrillThrough(args) {
    args.actionInfo.ReportName = "PersonalDetails";
    args.actionInfo.Parameters = [{ name: 'EmployeeID', value: ['3'] }];
}
</script>
```

Hyperlink

The **hyperlink** event occurs when users click a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
'html
```

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"  
report-path="Customer Support Analysis (Random data).rdl" hyperlink="onHyperlink">  
</bold-report-viewer>  
<script type="text/javascript">  
function onHyperlink(args) {  
    args.cancel = true;  
    //You can modify the URL here  
    window.open(args.actionInfo.Hyperlink);  
}  
</script>  
'
```

Handle post actions

Report processing actions are sent in an Ajax request to exchange data with the Web API service. You can handle post actions event to customize the Ajax requests.

AjaxBeforeLoad

AjaxSuccess

AjaxError

AjaxBeforeLoad

This event can be triggered before an Ajax request is sent to the Report Viewer Web API service. It allows you to set additional headers and custom data in the Ajax request. The following code sample demonstrates adding custom authorization header and passing default parameter values to service.

Add custom header to Ajax request

Initialize the **ajaxBeforeLoad** event in the script and add the authorization token to the **headers** property.

```
'js
```

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"  
report-path="sales-order-detail.rdl" ajax-before-load="onAjaxRequest"></bold-report-viewer>  
<script type="text/javascript">  
function onAjaxRequest(args) {  
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });  
}
```

```
}
```

```
</script>
```

```
'
```

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#).

Get the custom header value from the `HttpContext` header collection using the key name specified at client side.

```
'csharp
```

```
private Microsoft.Extensions.Primitives.StringValues authenticationHeader;
```

```
[HttpPost]
```

```
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
```

```
{
```

```
//jsonArray = jsonResult;
```

```
if (jsonResult != null)
```

```
{
```

```
//Get client side custom ajax header and store in local variable
```

```
HttpContext.Request.Headers.TryGetValue("Authorization", out authenticationHeader);
```

```
//Perform your custom validation here
```

```
if (authenticationHeader == "")
```

```
{
```

```
return new Exception("Authentication failed!!!");
```

```
}
```

```
else
```

```
{
```

```
return ReportHelper.ProcessReport(jsonResult, this, _cache);
```

```
}
```

```
}
```

```
return ReportHelper.ProcessReport(jsonResult, this, this._cache);
```

```
}
```

```
'
```

Perform your own action to validate the header values.

[Pass custom data in Ajax request](#)

Use the `data` property to set custom data to the server in the Ajax request. In the following code sample, parameter values are passed to the server side.

```
'js
```

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-path="sales-order-detail.rdl" ajax-before-load="onAjaxRequest"></bold-report-viewer>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
</script>
`
```

The custom data values are stored in the `customData` header key, you can store it to the local property. The following code sample stores parameter values, then the values are set to the report in the `OnReportLoaded` method.

```
'csharp
public string DefaultParameter = null;
private Microsoft.Extensions.Primitives.StringValues authenticationHeader;
[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
//jsonArray = jsonResult;
if (jsonResult != null)
{
if (jsonResult.ContainsKey("customData"))
{
//Get client side custom data and store in local variable. Here parameter values are sent.
DefaultParameter = jsonResult["customData"].ToString();
}
//Get client side custom ajax header and store in local variable
HttpContext.Request.Headers.TryGetValue("Authorization", out authenticationHeader);
//Perform your custom validation here
if (authenticationHeader == "")
{
return new Exception("Authentication failed!!!!");
}
}
```

Handle post actions AjaxSuccess

```
else
{
    return ReportHelper.ProcessReport(jsonResult, this, _cache);
}
}

return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    if (DefaultParameter != null)
    {
        //Set client side custom header data
        reportOption.ReportModel.Parameters =
JsonConvert.DeserializeObject<List<BoldReports.Web.ReportParameter>>(DefaultParameter);
    }
}
`
```

AjaxSuccess

To perform custom action or show user defined message, use the `ajaxSuccess` event on the successful Ajax request.

```
`js
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-path="sales-order-detail.rdl" ajax-before-load="onAjaxRequest" ajax-
success="onAjaxSuccess"></bold-report-viewer>
<script type="text/javascript">
function onAjaxRequest(args) {
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
    //Passing custom parameter data to server
    args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
function onAjaxSuccess(args) {
    //Perform your custom success message here
    alert("Ajax request success!!!!");
}
</script>
```

AjaxError

The `ajaxError` event is called, if an error occurred with the request, you can display the customized error detail in the event method.

`js

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-path="sales-order-detail.rdl" ajax-before-load="onAjaxRequest" ajax-
error="onAjaxFailure"></bold-report-viewer>

<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
function onAjaxFailure(args) {
alert("Status: " + args.status + "\n" +
"Error: " + args.responseText);
}
</script>
```

You can never have both an error and a success callback with a request.

Error logging in ASP.NET Core Report Viewer

If an error occurred in report processing, ASP.NET Core Report Viewer displays short messages about the error in view. You need to configure the `Controller` to save all logs, stack trace, and error information into a physical file location.

This section explains how to log the detailed error information to your ASP.NET Core application.

This section requires a ASP.NET Core Report Viewer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

In Solution Explorer, open the Report Viewer Controller file.

Inherit the `IReportLogger` interface and implement the interface methods.

`csharp

```
public class ReportViewerController : Controller, IReportController, IReportLogger
{
```

```
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType errorType)
{
    throw new NotImplementedException();
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string methodName, string className)
{
    throw new NotImplementedException();
}

```

```

Create a method in `ReportViewerController` to write the error text into application folder.

```
`csharp
internal void WriteLogs(string errorMessage)
{
 string filePath = Path.Combine(_hostingEnvironment.WebRootPath, "ErrorDetails.txt");
 using (StreamWriter writer = new StreamWriter(filePath, true))
 {
 writer.AutoFlush = true;
 writer.WriteLine(errorMessage);
 }
}
`
```

Invoke the newly created function in `.LogError` as below.

```
`csharp
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType errorType)
{
 WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string methodName, string className)
```

```
{
 WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:
 {3}", className, methodName, errorDetail, exception.StackTrace));
}
`
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final controller given as follows, you can replace it in your application.

```
`csharp
public class ReportViewerController : Controller, IReportController, IReportLogger
{
 // Post action for processing the RDL/RDLC report
 public object PostReportAction(Dictionary<string, object> jsonResult)
 {
 return ReportHelper.ProcessReport(jsonResult, this);
 }
 // Get action for getting resources from the report
 [System.Web.Http.ActionName("GetResource")]
 [AcceptVerbs("GET")]
 public object GetResource(string key, string resourcetype, bool isPrint)
 {
 return ReportHelper.GetResource(key, resourcetype, isPrint);
 }
 // Method that will be called when initialize the report options before start processing the report
 public void OnInitReportOptions(ReportViewerOptions reportOption)
 {
 // You can update report options here
 }
 // Method that will be called when reported is loaded
 public void OnReportLoaded(ReportViewerOptions reportOption)
 {
 // You can update report options here
 }
}
```

[Print report](#)[View report in print mode](#)

```
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
{
 WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string
methodName, string className)
{
 WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:
{3}", className, methodName, errorDetail, exception.StackTrace));
}

internal void WriteLogs(string errorMessage)
{
 // Error details text file path location
 string filePath = Path.Combine(_hostingEnvironment.WebRootPath, "ErrorDetails.txt");
 using (StreamWriter writer = new StreamWriter(filePath, true))
 {
 writer.AutoFlush = true;
 writer.WriteLine(errorMessage);
 }
}
}
```

## Print report

The Report Viewer provides print report option in the toolbar to print a copy of the report. The page setup dialog allows you to set the paper size or other page setup properties. To see print margins, click **Print Layout** on the toolbar.

You can set values in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values of the Page Setup dialog is based on the report properties set in the design view.

## [View report in print mode](#)

Print margins are displayed in the print layout only. To view report in print mode by default, set the **print-mode** property to true.

```
`js
```

```
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-
url="/api/ReportViewer" print-mode="true" processing-mode="Remote"></bold-report-viewer>
```

By default, the Report Viewer renders report in normal layout in which the print margins are not displayed.

### Print in new page

To open the print in a new tab of the current browser, set the `printOption` property to `NewTab`. By default, it shows the print dialog in the same page.

'js

```
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-
url="/api/ReportViewer" print-mode="true" print-option="NewTab" processing-
mode="Remote"></bold-report-viewer>
```

The pop-up blocker should be enabled for the page to open the print view in new tab.

### Set page orientation and paper size

You can specify the print page paper size and orientation at client-side to change the page setup properties by setting the `page-settings` property.

The following code example illustrates how to set page orientation and paper size in the Report Viewer for client side.

'html

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
page-settings="ViewBag.pageSettings">
</bold-report-viewer>
```

The following code example illustrates how to set page orientation and paper size in the Report Viewer for server side.

'csharp

```
public ActionResult Index()
{
 ViewBag.pageSettings = new BoldReports.Models.ReportViewer.PageSettings();
 ViewBag.pageSettings.Orientation = BoldReports.ReportViewerEnums.Orientation.Landscape;
 ViewBag.pageSettings.PaperSize = BoldReports.ReportViewerEnums.PaperSize.Letter;
 return View();
}
```

### Set report margin

To set margin values to the report page setup, use the `Margins` property and specify the value to top, right, bottom, and left.

The following code example illustrates how to set report margin in the Report Viewer for client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
page-settings="ViewBag.pageSettings">
</bold-report-viewer>
'
```

The following code example illustrates how to set report margin in the Report Viewer for server side.

```
'csharp
public ActionResult Index()
{
 ViewBag.pageSettings = new BoldReports.Models.ReportViewer.PageSettings();
 ViewBag.pageSettings.Margins = new BoldReports.Models.ReportViewer.Margins();
 ViewBag.pageSettings.Margins.Top = 0.5;
 ViewBag.pageSettings.Margins.Left = 0.5;
 ViewBag.pageSettings.Margins.Bottom = 0.5;
 ViewBag.pageSettings.Margins.Right = 0.5;
 return View();
}
'
```

The values set in the margin property is considered as inches input.

### Set page height and width

To set height and width values to the report page setup, use the **Height** and **Width** properties.

The following code example illustrates how to set page height and width in the Report Viewer for client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
page-settings="ViewBag.pageSettings">
</bold-report-viewer>
'
```

The following code example illustrates how to set page height and width in Report Viewer for server side.

```
'csharp
public ActionResult Index()
{
```

```
ViewBag.pageSettings = new BoldReports.Models.ReportViewer.PageSettings();
ViewBag.pageSettings.Height = 11.69;
ViewBag.pageSettings.Width = 8.27;
return View();
}
```

The values set in the height and width properties are considered as inches input.

### Print report with images

When the report has more images, the browser will send the report stream to the print dialog before the images are completely loaded. To load the print report stream with complete images, you should set the `EmbedImageData` property to true in `OnInitReportOptions` as shown in the following code.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.EmbedImageData = true;
}
`
```

Replace the following code sample in the client-side HTML file.

```
`js
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-path="Product Details.rdl" page-settings="ViewBag.pageSettings">
</bold-report-viewer>
`
```

In this tutorial, the `Product Details.rdl` report is used, and it can be downloaded from [here](#).

### External styles in report printing

While printing report, the external styles are used in the application overrides printable page style and prints output with incorrect alignments. To avoid the external script overriding, set the `isStyleLoad` property to false, which will print the page using only the Report Viewer styles.

```
`js
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
report-path="Product Details.rdl" report-print="onReportPrint">
</bold-report-viewer>
<script type="text/javascript">
function onReportPrint(args) {
 args.isStyleLoad = false;
}
`
```

```
}
```

```
</script>
```

```
,
```

### Show print progress

Report Viewer provides events to show the progress information when the printing takes long time to complete.

To show print progress, follow these steps:

Set the **print-progress-changed** in Report Viewer initialization.

Implement the function and add code samples to show a custom message based on the print progress status as shown in the following code snippet.

```
'js
```

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
```

```
report-path="Product Details.rdl" print-progress-changed="onPrintProgressChanged">
```

```
</bold-report-viewer>
```

```
<script type="text/javascript">
```

```
function onPrintProgressChanged(args) {
```

```
if (args.stage == "beginPrint") {
```

```
$('#viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing print data..
```

```
Please wait..." });
```

```
}
```

```
if (args.stage == "printStarted") {
```

```
var popupObj = $('#viewer').data('ejWaitingPopup');
```

```
popupObj.hide();
```

```
}
```

```
else if (args.stage == "preparation") {
```

```
console.log(args.stage);
```

```
if (args.preparationStage == "dataPreparation") {
```

```
console.log(args.preparationStage);
```

```
console.log(args.totalPages);
```

```
console.log(args.currentPage);
```

```
if (args.totalPages > 1 && args.currentPage > 1) {
```

```
var progressPercentage = Math.floor((args.currentPage / args.totalPages) * 100);
```

```
if (progressPercentage > 0) {
```

Export report

Remove empty spaces in printing

```
var popupObj = $('#viewer').data('ejWaitingPopup');
popupObj.setModel({ text: "Preparing print data.." + progressPercentage + " % completed.. Please
wait..." });
}
}
}
}

args.handled = true;
}
</script>
`
```

### Remove empty spaces in printing

The extra blank page is created when the body of your report is too wide for your page. To make the report appear on a single page, all the content within the report body must fit on the physical page, and the body width should be as the following formula.

Body Width <= Page Width - (Left Margin + Right Margin)

For more details about removing the empty pages in the report while designing, refer to the knowledge base article of [report page sizing](#).

## Export report

The Report Viewer provides events and properties to control and customize the report exporting functionality.

### Export event handling

You can show the progress information, when the exporting process takes long time to complete using the `export-progress-changed` event.

Set the `export-progress-changed` event in Report Viewer initialization.

Implement the function and replace the following code samples to show a custom message based on the progress stage.

The following code example demonstrates how to export event handling in the Report Viewer at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-progress-changed="onExportProgressChanged">
</bold-report-viewer>
<script type="text/javascript">
function onExportProgressChanged(args) {
```

```
if (args.stage === "beginExport") {
 console.log(args.stage);
 args.format =
 $('#viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing exporting document.. Please wait..." });
}

else if (args.stage === "exportStarted") {
 console.log(args.stage);
 var popupObj1 = $('#viewer').data('ejWaitingPopup');
 popupObj1.hide();
}

else if (args.stage === "preparation") {
 console.log(args.stage);
 console.log(args.format);
 console.log(args.preparationStage);
 if (args.format === "PDF" && args.preparationStage === "documentPreparation") {
 console.log(args.totalPages);
 console.log(args.currentPage);
 if (args.totalPages > 1 && args.currentPage > 1) {
 var progressPercentage = Math.floor((args.currentPage / args.totalPages) * 100);
 if (progressPercentage > 0) {
 var popupObj2 = $('#viewer').data('ejWaitingPopup');
 popupObj2.setModel({ text: "Preparing exporting document.." + progressPercentage + " % completed.. Please wait..." });
 }
 }
 }
 args.handled = true;
}
</script>
`
```

## Change Excel and Word export format

Allows you change the default file format to any other file format using the **ExcelFormat** and **WordFormat** properties.

The following code example demonstrates how to change Excel and Word export format in the Report Viewer at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-settings="ViewBag.exportSettings">
</bold-report-viewer>
'
```

The following code example demonstrates how to change Excel and Word export format in the Report Viewer at server side.

```
'csharp
public ActionResult Index()
{
 ViewBag.exportSettings = new BoldReports.Models.ReportViewer.ExportSettings();
 ViewBag.exportSettings.ExcelFormat = BoldReports.ReportViewerEnums.ExcelFormats.Excel2013;
 ViewBag.exportSettings.WordFormat = BoldReports.ReportViewerEnums.WordFormats.Word2013;
 return View();
}'
```

## Export data visualization items

To export the reports with data visualization components, it is mandatory to configure the web scripts in Report Viewer Web API controller. If the report definition uses chart, gauge and map report items then configure the scripts in Web API as the following steps,

Open the Report Viewer Web API controller.

Configure the below scripts and styles in **OnInitReportOptions** on Web API controller.

**jquery-1.10.2.min.js**

**bold.reports.common.min.js**

**bold.reports.widgets.min.js**

**ej.chart.min.js**

**ej2-base.min.js**

`ej2-data.min.js``ej2-pdf-export.min.js``ej2-svg-base.min.js``ej2-lineargauge.min.js``ej2-circulargauge.min.js``ej.map.min.js``bold.report-viewer.min.js`

You can replace the following codes in Report Viewer Web API controller.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
 {
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
 //Chart component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
 //Gauge component scripts
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
 //Map component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
 //Report Viewer Script
 "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
 };
}
```

```
reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
 "https://code.jquery.com/jquery-1.10.2.min.js"
};
```

The data visualization components will not export without the above script configurations.

#### Export data visualization items in core environment

Report Viewer uses **WebBrowser** to export the data visualization items to PDF, Word, Excel file formats. The **WebBrowser** is not supported in Core environment. To overcome this limitation in Core environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

It supported **ASP.NET Core 2.1 or above** **ASP.NET Core 2.1** version only.

To download **PhantomJS** application and deploy it on your machine, you should accept it's license terms on [LICENSE](#) and [Third-Party](#) document.

Download **PhantomJS** from [here](#) and extract the download file.

Copy the **PhantomJS.exe** file from the extracted bin folder and paste into **wwwroot/PhantomJS** folder in your application.

Open the Report Viewer Web API controller.

Set the **UsePhantomJS** property to true and **PhantomJSPath** property in **OnInitReportOptions** method.

```
'csharp
// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 string basePath = _hostingEnvironment.WebRootPath;
 reportOption.ReportModel.ExportResources.UsePhantomJS = true;
 reportOption.ReportModel.ExportResources.PhantomJSPath = basePath + @"\PhantomJS\";
 reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
 {
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
 };
}
```

Export report

Hide specific export type for report

```
//Chart component script
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
//Gauge component scripts
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
//Map component script
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
//Report Viewer Script
"https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
};

reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
 "https://code.jquery.com/jquery-1.10.2.min.js"
};
```

The **Scripts** and **Dependent** scripts must be added to export the items. For more details refer to the [export data visualization items](#) section.

#### Hide specific export type for report

Show or hide the default export types available in the component using the **ExportOptions** property.

The following code example demonstrates how to hide specific export type to the report in the Report Viewer at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-settings="ViewBag.exportSettings">
</bold-report-viewer>
'
```

The following code example demonstrates how to hide specific export type to the report in the Report Viewer at server side.

```
'csharp
```

```
public ActionResult Index()
{
 ViewBag.exportSettings = new BoldReports.Models.ReportViewer.ExportSettings();
 ViewBag.exportSettings.ExportOptions = BoldReports.ReportViewerEnums.ExportOptions.All
 & ~BoldReports.ReportViewerEnums.ExportOptions.Pdf;
 return View();
}
`
```

## PDF export options

The **PDFOptions** provides properties to manage PDF export behaviors. You should set the properties in the **OnInitReportOptions** method of the Web API service.

### Export with complex scripts

To export reports with the complex scripts, set the **ComplexScript** property of **PDFOptions** instance to true.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
 {
 EnableComplexScript = true
 };
}
`
```

## PDF conformance

You can export the report as a **PDF/A-1b** document by specifying the **PdfConformanceLevel.Pdf\_A1B** conformance level in the **PdfConformanceLevel** property.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
 {
 PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B
 };
}
`
```

### Add custom PDF fonts

You can add custom fonts to the PDF exported document by adding the font streams to **Fonts** collection in **PDFOptions** instance.

To add custom fonts to the PDF exported document, follow these steps:

Add the font .ttf files into your application **wwwroot/Resources** folder.

In the Solution Explorer, open the properties of the font file and set the **Copy** property to Output Directory as Copy always.

Initialize the **Font** collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

`'csharp`

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 string basePath = _hostingEnvironment.WebRootPath;
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
 {
 //Load Missing font stream
 Fonts = new Dictionary<string, System.IO.Stream>
 {
 { "Segoe UI", new FileStream(basePath + @"\Resources\font_symbols.ttf", FileMode.Open,
 FileAccess.Read) }
 }
 };
}
```

If any fonts used in the report definition is not installed or available in the local system, then you should load the font stream. In the above code, **font\_symbols** font stream is loaded to export the **sales-order-detail.rdl** report as symbols.

### Word export options

The **WordOptions** provides properties to manage Word document export behaviors.

#### Word document type

You can save the report to the required document version by setting the **FormatType** property.

`'csharp`

```
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
 FormatType = BoldReports.Writer.WordFormatType.Docx
};
`
```

### Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the **LayoutOption** to **TopLevel**. The **ParagraphSpacing** is the distance value added between two elements in the document.

```
`csharp
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
 LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel,
 ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()
 {
 Bottom = 0.5f,
 Top = 0.5f
 }
};
`
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, add an empty paragraph between two tables.

### Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without password. The following are the types of protection:

**AllowOnlyComments:** Adds or modifies only the comments in the Word document.

**AllowOnlyFormFields:** Modifies the form field values in the Word document.

**AllowOnlyRevisions:** Accepts or rejects the revisions in the Word document.

**AllowOnlyReading:** Views the content only in the Word document.

**NoProtection:** Accesses or edits the Word document contents as normally.

```
'csharp
reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
{
 ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading
};
```

## Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

### Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

```
'csharp
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
 ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013
};
```

### Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the **LayoutOption** to **IgnoreCellMerge**.

```
'csharp
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
 LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge
};
```

### Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the **ExcelSheetProtection** or enabling the **ReadOnlyRecommended** properties.

```
'csharp
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
 ReadOnlyRecommended = true,
 ExcelSheetProtection = Syncfusion.XlsIO.ExcelSheetProtection.DeletingColumns
};
```

## CSV export options

The `CsvOptions` allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
'csharp
```

```
reportOption.ReportModel.CsvOptions = new BoldReports.Writer.CsvOptions()
{
 Encoding = System.Text.Encoding.Default,
 FieldDelimiter = ",",
 UseFormattedValues = false,
 Qualifier = "#",
 RecordDelimiter = "@",
 SuppressLineBreaks = true,
 FileExtension = ".txt"
};
```

## Password protect exported document

Allows you protect the exported document such as PDF, Word, Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet illustrates how to encrypt the exported document with user-defined password.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 //PDF encryption
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions();
 reportOption.ReportModel.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
 {
 UserPassword = "password"
 };
 //Word encryption
 reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
 {
 EncryptionPassword = "password"
 };
}
```

```
//Excel encryption
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
 PasswordToModify = "password",
 PasswordToOpen = "password"
};
}
'
```

Password protection is not supported for HTML export format.

## Custom Actions

Add user defined buttons to the toolbar and invoke custom actions using the **Report Viewer** property. You can create a custom email button with the rendered report to users.

### Add email button

Create an email button in the toolbar using the **CustomItems** property with the values such as **GroupIndex**, **Index**, **Type**, **CssClass**, and **Tooltip**. The **tool-bar-item-click** event triggers when you click the email button.

Access the Report Viewer model and create a JSON array for sending requests to the Web API Server.

You can use the following codes to add email button from controller and passing the data to view using **ViewBag**.

```
'csharp
public ActionResult Index()
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 toolbarSettings.CustomItems = new List<CustomItem>();
 var customItem = new CustomItem()
 {
 GroupIndex = 1,
 Index = 1,
 CssClass = "e-icon e-mail e-reportviewer-icon",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "E-Mail",
```

```
Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
};
toolbarSettings.CustomItems.Add(customItem);
ViewBag.toolbarSettings = toolbarSettings;
return View();
}
`
```

You can use the following codes to set an **toolbar-settings** property at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">
</bold-report-viewer>
`
```

You can use the following codes to create an **tool-bar-item-click** event at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
tool-bar-item-click="onToolBarItemClick">
</bold-report-viewer>
<script type="text/javascript">
function onToolBarItemClick(args) {
 alert('Action Triggered');
}
</script>
`
```

You can use the following codes to get **toolbar-settings** properties on a dynamic object using **ViewBag.toolbarSettings** and invoke custom actions.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings" tool-bar-item-click="onToolBarItemClick">
</bold-report-viewer>
<script type="text/javascript">
function onToolBarItemClick(args) {
```

```
if (args.value == "E-Mail") {
 alert('Action Triggered');
}
}
</script>
'
```

## Toolbar customization

You can hide the component toolbar to show customized user interface or to customize the toolbar icons and element's appearances using the templates and Report Viewer toolbar customization properties.

In this tutorial, the `sales-order-detail.rdl` report is used and it can be downloaded from [here](#). You can add the reports from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

### Hide parameter block and toolbar items

To hide toolbar items, set the `toolbar-settings` property. The following code can be used to remove the parameter option from the toolbar and hide the parameter block.

The following code example demonstrates how to hide the parameter block in the Report Viewer at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">
</bold-report-viewer>
'
```

The following code example demonstrates how to hide the parameter block in the Report Viewer at server side.

```
'csharp
public ActionResult Index()
{
 ViewBag.toolbarSettings = new BoldReports.Models.ReportViewer.ToolbarSettings();
 ViewBag.toolbarSettings.Items = BoldReports.ReportViewerEnums.ToolbarItems.All
 & ~BoldReports.ReportViewerEnums.ToolbarItems.Parameters;
 return View();
}'
```

Similarly, you can show or hide all other toolbar options with the help of `toolbarSettings.Items` enum.

### Enable stop option in toolbar

To enable stop option in toolbar, set the `toolbarSettings.Items` property to `BoldReports.ReportViewerEnums.ToolbarItems.All`. The following code can be used to enable stop option in toolbar.

The following code example demonstrates how to enable stop option in the Report Viewer toolbar at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">
</bold-report-viewer>
'
```

The following code example demonstrates how to enable stop option in the Report Viewer toolbar at server side.

```
'csharp
public ActionResult Index()
{
 ViewBag.toolbarSettings = new BoldReports.Models.ReportViewer.ToolbarSettings();
 ViewBag.toolbarSettings.Items = BoldReports.ReportViewerEnums.ToolbarItems.All;
 return View();
}
'
```

### Hide toolbar

To hide the Report Viewer toolbar, set the `ShowToolbar` property to false.

The following code example demonstrates how to hide the toolbar in the Report Viewer at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">
</bold-report-viewer>
'
```

The following code example demonstrates how to hide the toolbar in the Report Viewer at server side.

```
'csharp
public ActionResult Index()
{
 ViewBag.toolbarSettings = new BoldReports.Models.ReportViewer.ToolbarSettings();
 ViewBag.toolbarSettings.ShowToolbar = false;
}
```

```
return View();
}
`
```

### Decide or hide the export option

The Report Viewer provides the `ExportOptions` property to show or hide the default export types available in the component. The following code hides the HTML export type from the default export options.

The following code example demonstrates how to decide or hide the export option in the Report Viewer at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-settings="ViewBag.exportSettings">
</bold-report-viewer>
`
```

The following code example demonstrates how to decide or hide the export option in the Report Viewer at server side.

```
'csharp
public ActionResult Index()
{
 ViewBag.exportSettings = new BoldReports.Models.ReportViewer.ExportSettings();
 ViewBag.exportSettings.ExportOptions = BoldReports.ReportViewerEnums.ExportOptions.All
 & ~BoldReports.ReportViewerEnums.ExportOptions.Html;
 return View();
}
`
```

### Add custom items to the export drop-down

To add custom items to the export drop-down available in the Report Viewer toolbar, use the property `CustomItems` and provide the JSON array of collection input with the `Index`, `CssClass` name, and `Value` properties.

You can use the following codes to add custom items to the export drop-down list from controller and passing the data to view using `ViewBag`.

```
'csharp
public ActionResult Index()
{
 ExportSettings exportSettings = new ExportSettings();
 exportSettings.CustomItems = new List<CustomExportItem>();
```

Toolbar customization

Add custom items to the export drop-down

```
var exportItem1 = new CustomExportItem() { Index = 2, CssClass = "", Value = "Text File" };
var exportItem2 = new CustomExportItem() { Index = 4, CssClass = "", Value = "DOT" };
exportSettings.CustomItems.Add(exportItem1);
exportSettings.CustomItems.Add(exportItem2);
ViewBag.exportSettings = exportSettings;
return View();
}
```

'

You can use the following codes to set an **export-settings** property at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-settings="ViewBag.exportSettings">
</bold-report-viewer>
'
```

You can use the following codes to create an **export-item-click** event at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-item-click="onExportItemClick">
</bold-report-viewer>
<script type="text/javascript">
//Export click event handler
function onExportItemClick(args) {
alert('Action Triggered');
}
</script>
'
```

You can use the following codes to get **export-settings** properties on a dynamic object using **ViewBag.exportSettings** and invoke custom actions.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
export-settings="ViewBag.exportSettings" export-item-click="onExportItemClick">
</bold-report-viewer>
<script type="text/javascript">
//Export click event handler

```

```
function onExportItemClick(args) {
if (args.value === "Text File") {
//Implement the code to export report as Text
alert("Text File export option clicked");
} else if (args.value === "DOT") {
//Implement the code to export report as DOT
alert("DOT export option clicked");
}
}

</script>
\
```

### Add custom toolbar item

You can add custom items to Report Viewer toolbar using the `toolbar-settings` property. You must register the `tool-bar-item-click` event to handle the newly added custom items action.

You can use the following codes to add custom toolbar item from controller and passing the data to view using `ViewBag`.

```
'csharp
public ActionResult Index()
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 toolbarSettings.CustomItems = new List<CustomItem>();
 var customItem = new CustomItem()
 {
 GroupIndex = 1,
 Index = 1,
 CssClass = "e-icon e-mail e-reportviewer-icon",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "E-Mail",
 Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
 };
 toolbarSettings.CustomItems.Add(customItem);
 ViewBag.toolbarSettings = toolbarSettings;
 return View();
}
```

You can use the following codes to set an **toolbar-settings** property at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">
</bold-report-viewer>
'
```

You can use the following codes to create an **tool-bar-item-click** event at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings" tool-bar-item-click="onToolBarItemClick">
</bold-report-viewer>
<script type="text/javascript">
function onToolBarItemClick(args) {
 alert('Action Triggered');
}
</script>
'
```

#### Add custom item to exiting toolbar group

To add a custom item to existing toolbar group use the property **CustomGroups** in **ToolbarSettings** and provide the JSON array of collection input with the **GroupIndex**, **Items**, **Type**, **CssClass** name, and **Tooltip** properties as given in following code snippet.

You can use the following codes to add custom item to exiting toolbar group from controller and passing the data to view using **ViewBag**.

```
'csharp
public ActionResult Index()
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 var groupItem = new List<CustomItem>();
 groupItem.Add(new CustomItem()
 {
 CssClass = "e-icon e-mail e-reportviewer-icon CustomGroup",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "CustomGroup",
 Tooltip = new ToolTip() { Header = "CustomGroup", Content = "toolbargroups" }
 };
 toolbarSettings.CustomGroups = groupItem;
 ViewBag.toolbarSettings = toolbarSettings;
}
```

```
});

groupItem.Add(new CustomItem()
{
 CssClass = "e-icon e-mail e-reportviewer-icon subCustomGroup",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "subCustomGroup",
 Tooltip = new ToolTip() { Header = "subCustomGroup", Content = "subtoolbargroups" }
});

toolbarSettings.CustomGroups.Add(new CustomGroup() { Items = groupItem, GroupIndex = 4 });

ViewBag.toolbarSettings = toolbarSettings;
return View();
}
```

You can use the following codes to set an `toolbar-settings` property at client side.

```
'html

<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">

</bold-report-viewer>
`
```

You can use the following codes to create an `tool-bar-item-click` event at client side.

```
'html

<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
tool-bar-item-click="onToolBarItemClick">

</bold-report-viewer>

<script type="text/javascript">
function onToolBarItemClick(args) {
 alert('Action Triggered');
}
</script>
`
```

You can use the following codes to get `toolbar-settings` properties on a dynamic object using `ViewBag.toolbarSettings` and invoke custom actions.

```
'html
```

```
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings" tool-bar-item-click="onToolBarItemClick">
</bold-report-viewer>
<script type="text/javascript">
function onToolBarItemClick(args) {
if (args.value === "CustomGroup") {
//Implement the code to CustomGroup toolbar option
alert("CustomGroup toolbar option clicked");
}
if (args.value === "subCustomGroup") {
//Implement the code to subCustomGroup toolbar option
alert("SubCustomGroup toolbar option clicked");
}
}
</script>
`
```

#### Add new toolbar group

To add new toolbar group and custom items to it, use the property `CustomItems` in `ToolbarSettings` and provide the JSON array of collection input with the `GroupIndex`, `Index` properties. The `CustomItem` must have the properties `Type`, `CssClass` and `Tooltip` as given in following code snippet.

You can use the following codes to add email button from controller and passing the data to view using `ViewBag`.

```
`csharp
public ActionResult Index()
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 toolbarSettings.CustomItems = new List<CustomItem>();
 var customItem = new CustomItem()
 {
 GroupIndex = 1,
 Index = 1,
 CssClass = "e-icon e-mail e-reportviewer-icon",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "E-Mail",
 }
}
```

```
Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
};
toolbarSettings.CustomItems.Add(customItem);
ViewBag.toolbarSettings = toolbarSettings;
return View();
}
`
```

You can use the following codes to set an `toolbar-settings` property on a dynamic object using `ViewBag.toolbarSettings` at client side.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" processing-mode="Remote"
toolbar-settings="ViewBag.toolbarSettings">
</bold-report-viewer>
`
```

## Localization of Bold Reports ASP.NET Core Report Viewer

Localization of ASP.NET Core Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. To render the static text with specific culture, refer to the following corresponding culture script files and set culture name to the `locale` property of the Report Viewer.

`ej.localetexts.fr-FR.min.js`

`ej.culture.fr-FR.min.js`

Refer to the `ej.localetexts.fr-FR.min.js` script file from the CDN link using the following code.

```
'html
<script src="http://cdn.boldreports.com/2.2.32/scripts/l10n/ej.localetexts.fr-FR.min.js"></script>
`
```

Refer to the `ej.localetexts.fr-FR.min.js` script file from the CDN link using the following code.

```
'html
<script src="http://cdn.boldreports.com/2.2.32/scripts/i18n/ej.culture.fr-FR.min.js"></script>
`
```

To render the localization Report Viewer, use the following code snippet.

The following code example illustrates how to localize Report Viewer in the index page.

```
'html
<bold-report-viewer id="viewer" report-service-url="/api/ReportViewer" locale="fr-FR"></bold-report-viewer>
`
```

The following code example illustrates how to localize Report Viewer in the layout page.

```
'html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Render Report Viewer in French localization</title>
<link
href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet"
/>
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/l10n/ej.localetexts.fr-FR.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/i18n/ej.culture.fr-FR.min.js"></script>
</head>
<body style="height:100%;width:100%;padding:0;">
<div class="container body-content" style="height:100%;width:100%;">
@RenderBody()
</div>
@RenderSection("scripts", required: false)
</body>
</html>
`
```

## Responsive layout rendering of ASP.NET Core Report Viewer

Report Viewer will adaptively render itself with optimal user interfaces for phone, tablet, or desktop form factors. This helps your application to scale elegantly on all form factors with ease. You can enable responsive layout rendering in Report Viewer by setting `isResponsive` property to true.

```
'html
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-
url="/api/ReportViewer" is-responsive="true">
</bold-report-viewer>
'
```

### Normal layout

The following output shows the normal layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in normal layout](/static/assets/aspnet-core/report-viewer/images/responsive-layout/normal-layout.png)

### Responsive layout

The following output shows the responsive layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/aspnet-core/report-viewer/images/responsive-layout/responsive-layout.png)

## Limitations

### RDL specification

The Report Viewer control does not support RDL specification for SQL Server 2000 and SQL Server 2005.

### Report layout

Vertical alignment in the text box report item is not supported in web rendering.

In the Tablix cell split layout process, the entire cell moves to the next page to display the complete cell items, when the table cell width value exceeds the page width.

### Expressions

The object function and VB function do not have complete support.

### SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you should specify the data source credentials for each report data source to establish database connection.

## Samples and Demos

Browse and explore the ready-to-use RDL, RDLC reports, samples, online, and offline demos.

### Locally installed reports

You can obtain the sample RDL and RDLC files from the Bold Reports installed location

%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.

### Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

### Online demos

You can view the ASP.NET Core Report Viewer online demo samples from [here](#).

### GitHub demo samples

Click [here](#) to view the GitHub Report Viewer demo samples.

## Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of ASP.NET Core Report Viewer application:

### Client-side migration

To uninstall NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search the **Syncfusion.EJ.ASPNET.Core** NuGet Package and uninstall from your application.

Search for **BoldReports.AspNet.Core** NuGet package, and install to your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

### Scripts and CSS references

Remove the following scripts and CSS references from the Report Viewer **\Views\Shared\\_Layout.cshtml** page:

**ej.web.all.min.css**

**ej.web.all.min.js**

Add the listed references in the same order given in above list. You can replace the following code in your **\Views\Shared\\_Layout.cshtml** page tag.

```
'html
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
```

```
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
<
```

The component prefix has been changed from ej to sf.

Open the \Views\Shared\\_Layout.cshtml page.

Remove the following code in your \_Layout.cshtml page.

Replace the following code in your \_Layout.cshtml page.

```
'html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>@ ViewData["Title"] - ReportViewerDemo</title>
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
</head>

<body>
<div style="min-height: 600px; width: 100%;">
 @RenderBody()
 </div>
 @RenderSection("Scripts", required: false)
 <!-- Bold Reports script manager -->
```

```
<bold-script-manager></bold-script-manager>
</body>
</html>
`
```

### Adding data visualization scripts

To render the report with data visualization components such as chart, gauge, and map items, add scripts of the visualization element. The following table shows the script reference that needs to be added in Report Viewer page for data visualization elements.

Visualization item | Script file

Chart | ej.chart.min.js|

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js, ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item, add chart control script ej.chart.min.js before the bold.report-viewer.min.js reference in the \Views\Shared\\_Layout.cshtml page as demonstrated in the following code sample.

```
'html
<link
href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet"
/>

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<!--Render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

The following code can be used to render the chart, gauge, and map report items in Report Viewer.

```
'html
<link
href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet"
/>
```

```

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>

<!--Render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>

<!--Render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>

<!--Render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>

<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

## Tag helper

Remove following tag helper within the `_ViewImports.cshtml` page.

```

`js
@using Syncfusion.JavaScript
@addTagHelper *, Syncfusion.EJ
`
```

Add following tag helper within the `_ViewImports.cshtml` page.

```

`js
@using BoldReports.TagHelpers
@addTagHelper *, BoldReports.AspNetCore
`
```

## Control initialization

The component prefix has been changed from `ej-report-viewer` to `bold-report-viewer`.

Open the Report Viewer CSHTML page.

Replace the component tag `ej-report-viewer` to `bold-report-viewer`.

```
'js
```

```
<bold-report-viewer id="viewer"></bold-report-viewer>
```

If your application contains Report Viewer initialization in multiple pages then replace in all the pages.

## Server-side migration

To uninstall NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search the `Syncfusion.EJ.ReportViewer.ASPNET.Core` NuGet Package and uninstall from your application.

Search for `BoldReports.Net.Core` NuGet package, and install to your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

## Web API Controller

The `IReportController` interface is moved to `BoldReports.Web.ReportViewer`. Open the Report Viewer Web API Controller file and remove the following using statement.

```
'csharp
```

```
using Syncfusion.EJ.ReportViewer;
```

Add the following using statement.

```
'csharp
```

```
using BoldReports.Web.ReportViewer;
```

Your application is successfully upgraded to the latest version of Report Viewer, and you can run the application with new assemblies.

## NuGet Packages for ASP.NET Core

Refer to the following steps to configure Bold Reporting NuGet packages for ASP.NET Core application.

## Configure NuGet feed URL

### Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** NuGet.org

**Source:** <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/aspnet-core/report-viewer/images/nuget-packages/NuGet\_Config.png)

### Offline NuGet feed URL

Bold Reporting NuGet packages are shipped into our Bold Reporting Tools build. To configure the packages from Bold Reports installed location, use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** Bold Reports installed NuGet

**Source:** {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages.

![Offline NuGet Configure](/static/assets/aspnet-core/report-viewer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

## Installing NuGet packages

### Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**. Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages (**BoldReports.Net.Core** or **BoldReports.AspNet.Core**), and then click **Install** button.

#### Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

#### install specified package in default project

```
Install-Package <Package Name>
```

#### install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

#### install specified package in specified project

```
Install-Package <Package Name> -ProjectName <Project Name>
```

For example:

```
'cmd
```

#### install specified package in default project

```
Install-Package BoldReports.Net.Core
```

#### install specified package in default project with specified Package Source

```
Install-Package BoldReports.Net.Core –Source “https://api.nuget.org/v3/index.json”
```

#### install specified package in specified project

```
Install-Package BoldReports.Net.Core -ProjectName BoldReportsApplication
```

#### Upgrading NuGet packages

##### Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**.  
Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

### Upgrading using Package Manager Console

To update the installed Bold Reporting NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

### Update specific NuGet package in default project

```
Update-Package <Package Name>
```

### Update all the packages in default project

```
Update-Package
```

### Update specified package in default project with specified package source

```
Update-Package <Package Name> -Source <Source Location>
```

### Update specified package in specified project

```
Update-Package <Package Name> - ProjectName <Project Name>
```

**For example:**

```
'cmd
```

### Update specified Bold Reporting NuGet package

```
Update-Package BoldReports.Net.Core
```

### Update specified package in default project with specified Package Source

```
Update-Package BoldReports.Net.Core –Source "https://api.nuget.org/v3/index.json"
```

### Update specified package in specified project

```
Update-Package BoldReports.Net.Core -ProjectName BoldReportsApplication
```

### Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing `nuget.exe` to latest version use the following command:

```
'cmd
nuget update -self
'
```

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting NuGet packages.

```
'cmd
```

### update all NuGet packages from config file

`nuget update <configPath> [options]`

### update all NuGet packages from specified Packages Source

`nuget update -Source <Source Location> [optional]`

`configPath` is optional. It identifies the `package.config` or solutions file lists the packages utilized in the project.

**For example:**

```
'cmd
```

### Update all NuGet packages from config file

`nuget update "C:\Users\BoldReportsApplication\package.config"`

### Update all NuGet packages from specified Packages Source

`nuget update -Source "https://api.nuget.org/v3/index.json"`

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

## How to Create RDL/RDLC Report

The following sections explain about how to create a new RDL/RDLC report using Bold Report Designer, Microsoft Report Builder and Visual Studio Report Server project template.

[Create a RDL report](#)

[Create a RDLC report](#)

[Change the exporting document file name based on the parameter](#)

[Change the connection string datasource dynamically](#)

[Disable the vertical scrollbar in parameter panel](#)

[Generate RDL and RDLC reports programmatically](#)

## Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

Bold Reports Report Designer.

Microsoft Report Builder.

Visual Studio Report Server project template.

### [Bold Reports Report Designer](#)

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

### [Microsoft SQL Report Builder](#)

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

### [Visual Studio Report Server template](#)

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

## [Create a RDLC report using business object data source](#)

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

### [Prerequisites](#)

Microsoft Visual Studio 2017 or higher

[Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

### Create business object class

Open Visual Studio from the File menu and select **New Project**.

Create project with class library type from the project type list.

![Add a new class library project](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/class-library-project.png)

Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
 public string ProdCat { get; set; }
 public string SubCat { get; set; }
 public string OrderYear { get; set; }
 public string OrderQtr { get; set; }
 public double Sales { get; set; }
}
```

Clean and build the application.

### Add an RDLC report

Right-click the project and click **Add > New Item**.

Search Report with new item and select **Report Wizard** to start the report creation with dataset selection.

Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

### Data source and table configuration wizard

Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/choose-data-source-type.png)

Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/select-data-objects.png)

In the DataSet Properties wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/arrange-table-fields.png)

Choose the table layout and click **Next**.

Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/aspnet-core/report-viewer/images/how-to/create-report/sales-report-design.png)

To render the RDLC using Report Viewer, refer to the [RDLC Report](#) section.

## Render data visualization report items

To render the report with data visualization components such as chart, gauge and map items, must add scripts of the visualization element. The following table shows the script reference that need to be added in Report Viewer page for data visualization elements.

Visualization Item | Script File

Chart | ej.chart.min.js

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item, add chart control script ej.chart.min.js before the bold.report-viewer.min.js reference in \Views\Shared\\_Layout.cshtml page as in following code sample.

'html

```
<link href="~/Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
```

```

<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="~/Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="~/Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

The following code can be used to render the chart, gauge and map report items in Report Viewer.

```

`html
<link href="~/Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="~/Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script, only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
<!--Used to render the map item. Add this script, only if your report contains the map report item.-->
<script src="~/Scripts/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="~/Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

## How to use the custom code with Report Viewer

Custom code in a report allows to include new custom constants, variables, functions, or subroutines. As the VBCodeProvider is not available in .NET Core platform, report with custom codes requires additional

settings to enable this feature. This section describes the steps required to use custom codes with Report Viewer.

## Create a custom code class file

Create a new C# class file in your application.

The namespace should be **BoldReports.Processing.Helper** and class declaration should be **public static class Code** .

Write the C# methods equivalent to your report VB codes.

Here is the example code to convert the value to USD.

```
'csharp
namespace BoldReports.Processing.Helper
{
 public static class Code
 {
 public static string PrintHelloWorld()
 {
 return "Hello World";
 }
 }
}'
```

## Create a assembly reference and add it to Report Viewer

If you have created the custom code class, then you need to add the assemblies in the list with what you have used in your application to Report Viewer as shown in following code example.

In our Report Viewer application, assemblies needs to be added before calling the OnReportLoaded method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 string basePath = _hostingEnvironment.WebRootPath;
 reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;
 FileStream inputStream = new FileStream(basePath + @"\Resources\Product List.rdlc", FileMode.Open,
 FileAccess.Read);
 reportOption.ReportModel.Stream = inputStream;
```

```

reportOption.ReportModel.Assemblies.Add(this.GetType().GetTypeInfo().Assembly);
}
'
```

## Display ssrs rdl report in Bold Reports ASP.NET Core Report Viewer

This section explains you the steps required to create your first ASP.NET Core reporting application to display already created SSRS RDL report in Bold Reports ASP.NET Core Report Viewer without using a Report Server.

### Create ASP.NET Core application

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

Choose the **ASP.NET Core version** and select the Web Application **Model-View-Controller** template and then click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/aspnet-core/report-viewer/images/getting-started/aspnet-core-web-application-template.png)

### Enable Docker support

We are using `System.Drawing` to measure text size for `CanGrow` feature support with `Textbox` `ReportItem` it requires native `libgdiplus` library but which doesn't contain in default `microsoft/dotnet` image. So, if we select **Enable Docker Support** then we must add native `libgdiplus` library install command with `Dockerfile`.

```

`command
install System.Drawing native dependencies
RUN apt-get update \
&& apt-get install -y --allow-unauthenticated \
libc6-dev \
libgdiplus \
libx11-dev \
&& rm -rf /var/lib/apt/lists/*
`
```

The sample docker file can be downloaded from [here](#)

### List of dependency libraries

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command and then search for `BoldReports.AspNet.Core` and `BoldReports.Net.Core` packages, and

install them in your Core application. The following table provides details about the packages and their usage.

#### Package | Purpose

**Syncfusion.Compression.Net.Core** | Exports the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core**, and **Syncfusion.XlsIO.Net.Core** packages.

**Syncfusion.Pdf.Net.Core** | Exports the report to a PDF.

**Syncfusion.DocIO.Net.Core** | Exports the report to a Word.

**Syncfusion.XlsIO.Net.Core** | Exports the report to an Excel.

**Syncfusion.OfficeChart.Net.Core** | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

**Newtonsoft.Json** | Serializes and deserialize data for the Report Viewer. It is a mandatory package for Report Viewer, and the package version should be higher than 10.0.1 for NET Core 2.0 and others should be higher than 9.0.1.

**System.Data.SqlClient** | This is an optional package for Report Viewer. It should be referenced in project when the RDL report renders visual data from the SQL Server or SQL Azure data source based on RDL design. The package version should be higher than 4.1.0.

#### Refer scripts and CSS

Directly refer all the required scripts and style sheets from [CDN](#) links.

The following scripts and style sheets are mandatorily required to use the Report Viewer.

**bold.reports.all.min.css**

**jquery-1.10.2.min.js**

**bold.reports.common.min.js**

**bold.reports.widgets.min.js**

**ej.chart.min.js** - Renders the chart item. Add this script, only if your report contains the chart report item.

**ej2-base.min.js**, **ej2-data.min.js**, **ej2-pdf-export.min.js**, **ej2-svg-base.min.js**, **ej2-lineargauge.min.js** and **ej2-circulargauge.min.js** - Render the gauge item. Add these scripts only if your report contains the gauge report item.

**ej.map.min.js** - Renders the map item. Add this script only if your report contains the map report item.

**bold.report-viewer.min.js**

Open the **\Views\Shared\\_Layout.cshtml** page.

Add the listed references in the same order given in above list. You can replace the following code in your `\Views\Shared\_Layout.cshtml` page tag.

```
'html
<link
 href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css" rel="stylesheet"
/>

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
 type="text/javascript"></script>

<script
 src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
 src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>

<!--Render the chart item. Add this script only if your report contains the chart report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>

<!--Render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>

<!--Render the map item. Add this script only if your report contains the map report item.-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>

<!-- Report Viewer component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
`
```

To learn more about rendering a report with data visualization report items, refer to the [how to render data visualization report items](#) section.

### Tag helper

It is necessary to define the following tag helper within the `_ViewImports.cshtml` page to initialize the Report Viewer component with the tag helper support.

```
'js
@using BoldReports.TagHelpers
@addTagHelper *, BoldReports.AspNetCore
```

## Configure Script Manager

Open the `~/Views/Shared/_Layout.cshtml` page and add the reporting Script Manager at the end of `<body>` element as in the following code sample.

```
'html
<body>
<div style="min-height: 600px; width: 100%;">
@RenderBody()
</div>
@RenderSection("Scripts", required: false)
<!-- Bold Reports script manager -->
<bold-script-manager></bold-script-manager>
</body>
'
```

## Initialize Report Viewer

Initialize the Report Viewer as shown in the following code sample in your Report Viewer CSHTML page.

For an example, the `Index.cshtml` page can be replaced with the following code by removing the existing codes.

```
'html
<bold-report-viewer id="viewer"></bold-report-viewer>
'
```

## Add already created reports

The Report Viewer is only for rendering the reports. You must use a report generation tool to create a report and to learn more about this, refer to the [create RDL report](#) section.

Create a folder `Resources` into the `wwwroot` folder in your application to store the RDL reports.

Add already created reports to the newly created folder.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#). You can add the reports from Bold Reports installation location. For more information, refer to [samples and demos](#) section.

## Configure Web API

The interface `IReportController` has declaration of action methods that are defined in the Web API Controller for processing the RDL, RDLC, and SSRS reports and for handling request from the Report Viewer control. The `IReportController` has the following action methods declaration:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportViewController.cs**

![Adding a new controller to the project](/static/assets/aspnet-core/report-viewer/images/getting-started/add-aspnet-core-api-controller.png)

Click **Add**.

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewController** and add the following using statement.

```
'csharp
using System.IO;
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and then implement its methods.

Create local references for the interfaces given in following table.

Interface | Purpose

**IMemoryCache** | Report Viewer requires a memory cache to store the information of consecutive client request and have the rendered report viewer information in server.

**IHostingEnvironment** | IHostingEnvironment used to get the report stream from application **wwwroot\Resources** folder.

You cannot load the application report with path information in ASP.NET Core service. So, you should load the report as stream in **OnInitReportOptions**.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 string basePath = _hostingEnvironment.WebRootPath;
 // Here, we have loaded the sales-order-detail.rdl report from application the folder
 // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
'
```

```
FileStream reportStream = new FileStream(basePath + @"\Resources\" +
reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);
reportOption.ReportModel.Stream = reportStream;
}
`
```

You can replace the template code with the following code.

```
`csharp
[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
 // Report viewer requires a memory cache to store the information of consecutive client request and
 // have the rendered Report Viewer information in server.
 private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
 // IHostingEnvironment used with sample to get the application data from wwwroot.
 private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
 // Post action to process the report from server based json parameters and send the result back to the
 // client.
 public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
 Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
 {
 _cache = memoryCache;
 _hostingEnvironment = hostingEnvironment;
 }
 // Post action to process the report from server based json parameters and send the result back to the
 // client.
 [HttpPost]
 public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
 {
 return ReportHelper.ProcessReport(jsonArray, this, this._cache);
 }
 // Method will be called to initialize the report information to load the report with ReportHelper for
 // processing.
 public void OnInitReportOptions(ReportViewerOptions reportOption)
 {
```

```

string basePath = _hostingEnvironment.WebRootPath;
// Here, we have loaded the sales-order-detail.rdl report from application the folder
wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
FileStream reportStream = new FileStream(basePath + @"\Resources\" +
reportOption.ReportModel.ReportPath, FileMode.Open, FileAccess.Read);
reportOption.ReportModel.Stream = reportStream;
}

// Method will be called when reported is loaded with internally to start to layout process with
ReportHelper.

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

//Get action for getting resources from the report
[ActionName("GetResource")]
[AcceptVerbs("GET")]

// Method will be called from Report Viewer client to get the image src for Image report item.
public object GetResource(ReportResource resource)
{
 return ReportHelper.GetResource(resource, this, _cache);
}

[HttpPost]
public object PostFormReportAction()
{
 return ReportHelper.ProcessReport(null, this, _cache);
}
}
`
```

#### Enable cross-origin requests

Browser security prevents the Report Viewer from making requests to your Web API Service when both server-side and client-side requests run in different domains. To allow access to your Web API service from a different domain, enable the cross-origin requests.

Call `AddCors` in `Startup.ConfigureServices` to add CORS services to the app's service container. Replace the following code to allow any origin requests.

```

`csharp
public void ConfigureServices(IServiceCollection services)
```

```
{
services.AddMvc();
services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
{
 builder.AllowAnyOrigin()
 .AllowAnyMethod()
 .AllowAnyHeader();
}));}
}
```

To specify the CORS policy for home controller, add the [EnableCors] attribute to the controller class and specify the policy name.

```
'csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
public class ReportViewerController : Controller, IReportController
{
 public IActionResult Index()
 {
 return View();
 }

}
```

### Set report path and service URL

To render the reports available in the application, set the report-path and report-service-url properties of the Report Viewer. You can replace the following code in your Report Viewer page.

```
'html
<bold-report-viewer id="viewer" report-path="sales-order-detail.rdl" report-service-
url="/api/ReportViewer"></bold-report-viewer>
'
```

The report path property is set to the RDL report that is added to the project Resources folder.

### Preview the report

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Sales Order Detail Report](/static/assets/aspnet-core/report-viewer/images/getting-started/sales-order-detail.png)

#### See Also

[Render report with data visualization report items](#)

[Render Report Server reports](#)

[Create RDLC report](#)

[Render RDLC reports](#)

[Preview report in print mode](#)

[Set data source credential for shared data sources](#)

[Change data source connection string](#)

## How to change the exporting document file name based on parameter

Find the following steps to change the file based on parameter values in Report.

Create the file exporting file name using the parameters in **onRenderingComplete** event and store it in local variable.

```
'html
function onRenderingComplete(event) {
 var parameters = event.reportParameters;
 if(parameters){
 for (var i = 0; i < parameters.length; i++) {
 if(parameters[i].Name == "Department"){
 this.exportFileName = "Sales for " + parameters[i].Value;
 }
 }
 }
}'
```

Use the file Name property with export, click event to change the file using the value stored in local variable used for having the file using the parameters.

```
'html
function onExportItemClick(event) {
 event.fileName = this.exportFileName ;
}
'
```

## How to change the connection string datasource dynamically

Find the following steps to change the connection string in datasource.

You need a report action information to get the data source information from report. So, stored the JsonResult information to local property in **PostReportAction** method as shown in the following code example.

```
'csharp
private Dictionary<string, object> _jsonResult;
//Post action for processing the rdl/rdlc report
public object PostReportAction(Dictionary < string, object > jsonResult)
{
if (jsonResult != null && jsonResult.Keys.Count > 0)
{
_jsonResult = jsonResult;
}
return ReportHelper.ProcessReport(jsonResult, this);
}
'
```

Use the JsonResult information with **ReportHelper.GetDatasource** API to get the data source details of the reports and you have to use the **DataSourceCredentials** to change the connection string of the data source.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (jsonResult != null && jsonResult.Keys.Count > 0)
{
List<DataSourceInfo> datasources = ReportHelper.GetDataSources(jsonResult, this, cache);
foreach (DataSourceInfo item in datasources)
{
if (item.DataProvider == "SQL")
{
string connectionString = "Data Source = dataplatformdemodata.syncfusion.com; Initial Catalog = AdventureWorks; User ID = 'demoreadonly@data-platform-demo'; Password = 'N@c)=Y8s*1&dh'";
DataSourceCredentials DataSourceCredentials = new DataSourceCredentials();
}
```

```
DataSourceCredentials.Name = item.DataSourceName;
DataSourceCredentials.UserId = null;
DataSourceCredentials.Password = null;
DataSourceCredentials.ConnectionString = connectionString;
DataSourceCredentials.IntegratedSecurity = false;//if windows credentials means we need to pass true
as IntegratedSecurity
reportOption.ReportModel.DataSourceCredentials = new List<DataSourceCredentials>
{
 DataSourceCredentials
};
}
}
}
}
}
`
```

## How to disable the vertical scrollbar in parameter panel

To disable the vertical scrollbar in parameter panel, set the `enableparameterblockscroller` property to false.

```
Example
'js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
 enableParameterBlockScroller: false
});
</script>
`
```

## How to generate the RDL and RDLC reports programmatically with ASP.NET Core

The Bold Reports reporting library allows you to generate and preview RDL and RDLC reports programmatically with ASP.NET Core. The `ReportDefinition` class is defined as an object model of a report. You can create, add, and modify the report properties, report sections like header and footer and report items in the report definition instance.

The following steps illustrates how to create a basic report object model:

### Initialize a report definition

The following code snippet guides you in initializing the report definition.

```
'csharp

ReportDefinition CreateReport()
{
 ReportDefinition report = new ReportDefinition();
 report.ReportSections = new ReportSections();
 var reportSection = new ReportSection();
 report.ReportSections.Add(reportSection);
 reportSection.Width = new BoldReports.RDL.DOM.Size("6in");
 report.ReportUnitType = "Inch";
 report.RDLType = RDLType.RDL2010;
 return report;
}

BoldReports.RDL.DOM.Style AddStyle()
{
 BoldReports.RDL.DOM.Style style = new BoldReports.RDL.DOM.Style();
 style.Border = new BoldReports.RDL.DOM.Border();
 style.Border.Width = new BoldReports.RDL.DOM.Size("1pt");
 style.Border.Style = "Solid";
 style.Border.Color = "Black";
 return style;
}
'
```

### Create a report page header

The following code snippet guides you in creating a **PageHeader** report section and setting values to their properties.

```
'csharp

ReportDefinition CreateReport()
{
 ...
 ...
 ...
}
```

```
reportSection.Page = new BoldReports.RDL.DOM.Page();
reportSection.Page.Style = AddStyle();
reportSection.Page.PageHeight = "4in";
reportSection.Page.PageWidth = "6in";
reportSection.Page.PageHeader = CreateHeader();

...
...
...

}

PageHeader CreateHeader()
{
 PageHeader pageHeader = new PageHeader();
 pageHeader.Height = new BoldReports.RDL.DOM.Size("0.59167in");
 pageHeader.Style = AddStyle();
 pageHeader.PrintOnFirstPage = true;
 pageHeader.PrintOnLastPage = true;
 pageHeader.ReportItems = new ReportItems();
 var textBox = CreateTextBox("TextBox2", "Header", "1.61333in", "0.13833in", "2.2in", "0.34167in");
 pageHeader.ReportItems.Add(textBox);
 return pageHeader;
}
`
```

### Create a report page footer

The following code snippet guides you in creating a **PageFooter** report section and setting values to their properties.

```
`csharp
ReportDefinition CreateReport()
{
...
...
...
reportSection.Page.PageHeader = CreateFooter();
...
...
}
```

How to generate the RDL and RDLC reports programmatically with ASP.NET Core Initialize a report body section

```
...
}

PageFooter CreateFooter()
{
 PageFooter pageFooter = new PageFooter();
 pageFooter.Style = AddStyle();
 pageFooter.Height = new BoldReports.RDL.DOM.Size("0.59167in");
 pageFooter.ReportItems = new ReportItems();
 pageFooter.PrintOnFirstPage = true;
 pageFooter.PrintOnLastPage = true;
 var textBox = CreateTextBox("TextBox3", "Footer", "1.61333in", "0.05416in", "2.2in", "0.34167in");
 pageFooter.ReportItems.Add(textBox);
 return pageFooter;
}
`
```

## Initialize a report body section

Initialize a report body object and set the values to their properties like height, styles, and report items as shown in the following code snippet.

```
`csharp
ReportDefinition CreateReport()
{
 ...
 ...
 ...
 reportSection.Body = CreateBody();
 ...
 ...
 ...
}

Body CreateBody()
{
 var body = new Body();
 body.Height = new BoldReports.RDL.DOM.Size("2.03333in");
```

How to generate the RDL and RDLC reports programmatically with ASP.NET Core Initialize a report body section

```
body.Style = AddStyle();
body.ReportItems = new ReportItems();
var textBox = CreateTextBox("TextBox1", "Body", "1.58833in", "0.66333in", "2.225in", "0.59167in");
body.ReportItems.Add(textBox);
return body;
}
```

Using the following code snippets, you can create a **Textbox** report item and assign the values for their properties like name, styles, paragraphs, and dimension values.

```
`csharp
PageHeader CreateHeader()
{
...
...
...
pageHeader.ReportItems = new ReportItems();
var textBox = CreateTextBox("TextBox2", "Header", "1.61333in", "0.13833in", "2.2in", "0.34167in");
pageHeader.ReportItems.Add(textBox);

...
...
...
}
PageFooter CreateFooter()
{
...
...
...
...
pageFooter.ReportItems = new ReportItems();
var textBox = CreateTextBox("TextBox3", "Footer", "1.61333in", "0.05416in", "2.2in", "0.34167in");
pageFooter.ReportItems.Add(textBox);

...
...
...
}
```

How to generate the RDL and RDLC reports programmatically with ASP.NET Core Initialize a report body section

```
}

Body CreateBody()
{
...
...
...
body.ReportItems = new ReportItems();
var textBox = CreateTextBox("TextBox1", "Body", "1.58833in", "0.66333in", "2.225in", "0.59167in");
body.ReportItems.Add(textBox);
...
...
...
}

BoldReports.RDL.DOM.TextBox CreateTextBox(string name, string text, string left, string top, string width, string height)
{
 var textBox = new BoldReports.RDL.DOM.TextBox();
 textBox.Name = name;
 textBox.Height = new BoldReports.RDL.DOM.Size(height);
 textBox.Width = new BoldReports.RDL.DOM.Size(width);
 textBox.Left = new BoldReports.RDL.DOM.Size(left);
 textBox.Top = new BoldReports.RDL.DOM.Size(top);
 textBox.Style = new BoldReports.RDL.DOM.Style();
 textBox.Style.TextAlign = "Center";
 textBox.Style.VerticalAlign = "Top";
 textBox.Paragraphs = new Paragraphs();
 BoldReports.RDL.DOM.Paragraph paragraph = new BoldReports.RDL.DOM.Paragraph();
 TextRuns runs = new TextRuns();
 TextRun run = new TextRun();
 run.Style = new BoldReports.RDL.DOM.Style();
 run.Style.FontStyle = "Default";
 run.Style.TextAlign = "Center";
 run.Style.FontFamily = "Arial";
```

```
run.Style.FontSize = new BoldReports.RDL.DOM.Size("10pt");
run.Value = text;
runs.Add(run);
paragraph.Style = new BoldReports.RDL.DOM.Style();
paragraph.Style.VerticalAlign = "Top";
paragraph.Style.TextAlign = "Center";
paragraph.TextRuns = runs;
textBox.Paragraphs.Add(paragraph);
return textBox;
}
```

### Render the generated report in ReportViewer

You can render the report using the ReportViewer, after the report definition is created. The following code snippet illustrates how to render the report using the ReportViewer by report definition.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 var report = CreateReport();
 reportOption.ReportModel.ReportDefinition = reportDefinition;
}
```

## Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports ASP.NET Core Report Viewer.

[How can improve the performance and handle the large amounts of data with Report Viewer?](#)

[Is Report Viewer compatible with latest version of JQuery library?](#)

[Is the back action from drillthrough report will load the report again with Report Viewer?](#)

[Is Report Viewer supports with .NET Core on Linux Docker?](#)

[Is possible to change the culture of the date time parameter?](#)

[Is it possible to hide report parameters?](#)

[Is it possible to hide the export options in Report Viewer?](#)

Is possible to change the culture of the date time parameter  
ReportViewer

Render the generated report in

### [Is it possible to load reports providing parameter values at runtime?](#)

## Is possible to change the culture of the date time parameter

Yes, we can change the culture of the date time parameter for Report Viewer by changing the locale. You can make use the following reference to change the locale of Report Viewer.

### [ReportViewer Localization](#)

In Report Viewer, we could not change the culture of specific parameter or UI. So, we have to achieve the requirements only by changing the culture.

## Is it possible to hide the parameters in Report Viewer

Yes, it is possible to hide the parameters in Report Viewer. On hiding the parameters, the users will not be able to have the parameter block in the Report Viewer. Refer to this [Hide parameter block and toolbar items](#) section.

## Is it possible to hide the export options in Report Viewer

Yes, it is possible to hide the export options in Report Viewer. The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. Refer to this [Decide or Hide the export options](#) section.

## Is it possible to load reports providing parameter values at runtime

Yes, it is possible to load reports with application inputs as parameters in Report Viewer. You need to provide the input values to the Report Viewer from client side at runtime using the `parameters` property, which accepts JSON array values. Refer to this [Set parameter at client](#) section.

## Bold Report Writer

Report Writer is a class library that enables the user to render reports defined in Microsoft's RDL format (2008 or 2008 R2) as PDF, Word, Excel or CSV documents.

The important features of WPF Report Writer are listed as follows:

RDL Specification - Supports RDL specification for the SQL Server 2008 and RDL specification for the SQL Server 2008 R2 only. List of available report definition formats:  
[msdn.microsoft.com/library/dd297486\(SQL.100\).](http://msdn.microsoft.com/library/dd297486(SQL.100).)

Data sources - You can use advanced database servers Data Sources in Report Writer (SQL and Oracle).

Charts - Show all basic types of charts that are available in Microsoft RDL reports.

Tablix - Shows the summaries and simple tables.

Gauge - Shows measurement values by using expression values.

Textbox - Shows textbox data with expression support.

Export - Export report as PDF, Word, Excel and CSV.

Report Parameter - Views the report based on the report parameter value.

## Export SSRS RDL Report in Bold Reports ASP.NET Core Report Writer

The Report Writer is a class library that is used to export the RDL report with popular file formats like PDF, Microsoft Word, Microsoft CSV, and Microsoft Excel without previewing the report in webpage. This section describes how to export the RDL report in ASP.NET Core application using the **Report Writer**.

Bold Reports ASP.NET Core Report Writer works in **ASP .NET Core 2.1**, **ASP.NET Core 2.2**, and **ASP.NET Core 3.x** versions.

### Create ASP.NET Core application

Start Visual Studio 2019 and click **Create new project**.

Choose **ASP.NET Core Web Application**, and then click **Next**.

Change the project name, and then click **Create**.

In the dropdown with the **ASP.NET Core version**, choose **ASP.NET Core 2.1**.

Select the **Web Application (Model-View-Controller)** template, then click **Create**.

![Creating a new ASP.NET Core Application Project](/static/assets/aspnet-core/report-writer/images/getting-started/aspnet-core-web-application-template.png)

### List of dependency libraries

In the Solution Explorer tab right-click the project or solution , and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **BoldReports.Net.Core** and **System.Data.SqlClient** packages, and install them in your Core application. The following table provides details about the packages and their usage.

#### Package | Purpose

**Syncfusion.Compression.Net.Core** | Exports the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core**, and **Syncfusion.XlsIO.Net.Core** packages.

**Syncfusion.Pdf.Net.Core** | Exports the report to a PDF.

**Syncfusion.DocIO.Net.Core** | Exports the report to a Word.

**Syncfusion.XlsIO.Net.Core** | Exports the report to an Excel.

**Syncfusion.OfficeChart.Net.Core** | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

**Newtonsoft.Json** | Serializes and deserialize data for the Report Writer. It is a mandatory package for Report Writer, and the package version should be 10.0.1 or higher.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded [here](#). You can get the reports from Bold Reports installation location. For more information, refer to [samples and demos](#) section.

## Server side Report Writer changes

Create a folder `Resources` into the `wwwroot` folder in your application. Copy and paste the sample RDL reports into the `Resources` folder.

Create local variables inside the `HomeController` class.

```
'csharp
// IHostingEnvironment used with sample to get the application data from wwwroot.
private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
// IHostingEnvironment initialized with controller to get the data from application data folder.
public HomeController(Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
{
 _hostingEnvironment = hostingEnvironment;
}
'
```

Open the `HomeController.cs` file in your application and add the `Export()`function to load the report as stream. Refer the following code snippet,

```
'csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
 // Here, we have loaded the sales-order-detail sample report from application the folder
 // wwwroot\Resources.

 FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath + @"\Resources\sales-
 order-detail.rdl", FileMode.Open, FileAccess.Read);

}
```

Initialize the Report Writer instance with report stream and set the specified export format and file name to the export document.

```

`csharp
public IActionResult Export(string writerFormat)
{

 BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);
 string fileName = null;
 WriterFormat format;
 string type = null;
 fileName = "sales-order-detail.pdf";
 type = "pdf";
 format = WriterFormat.PDF;
}
```

```

You can use the `Save` method in Report Writer to generate the export document along with information of the report stream, it will return the generated file as Stream.

```

`csharp
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
```

```

Refer to the following complete code snippet used to get the exported file stream.

```

`csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
 // Here, we have loaded the sales-order-detail sample report from application the folder
 // wwwroot\Resources.

```

```
FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath + @"\Resources\sales-order-detail.rdl", FileMode.Open, FileAccess.Read);
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);
string fileName = null;
WriterFormat format;
string type = null;
if (writerFormat == "PDF")
{
 fileName = "sales-order-detail.pdf";
 type = "pdf";
 format = WriterFormat.PDF;
}
else if (writerFormat == "Word")
{
 fileName = "sales-order-detail.doc";
 type = "doc";
 format = WriterFormat.Word;
}
else if (writerFormat == "CSV")
{
 fileName = "sales-order-detail.csv";
 type = "csv";
 format = WriterFormat.CSV;
}
else
{
 fileName = "sales-order-detail.xls";
 type = "xls";
 format = WriterFormat.Excel;
}
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
```

```
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
,
```

### Client side changes

Use the following code snippet in `Index.cshtml` home page to invoke the Web API from client side.

```
'html
{@Html.BeginForm("Export", "Home", FormMethod.Post);
{
<div>
<input type="submit" value="Generate" style="width: 150px;" />
</div>
}
Html.EndForm();
}
,
```

You can add the export file types to the home page to choose the format you want to export in the Report Writer. Copy and paste the following code snippet in your application.

```
'html
{@Html.BeginForm("Export", "Home", FormMethod.Post);
{
<div class="Common">
<div class="tablediv">
<div class="rowdiv">
<label id="design">
Choose the any one of the format to export the document in Report Writer.

</label>
</div>
```

```
<div class="rowdiv">
<div class="celldiv" style="padding:10px">
<label>
 Save As :
</label>
<input id="rbtnPDF" type="radio" name="writerFormat" value="PDF" checked="checked" style="margin-left: 15px" />
<label for="rbtnPDF" style="padding:0px 5px 0px 2px">
 PDF
</label>
<input id="rbtnWord" type="radio" name="writerFormat" value="Word" style="margin-left: 15px" />
<label for="rbtnWord" style="padding:0px 5px 0px 2px">
 Word
</label>
<input id="rbtnxls" type="radio" name="writerFormat" value="xls" style="margin-left: 15px" />
<label for="rbtnxls" style="padding:0px 5px 0px 2px">
 Excel
</label>
<input id="rbtnCSV" type="radio" name="writerFormat" value="CSV" style="margin-left: 15px" />
<label for="rbtnCSV" style="padding:0px 25px 0px 2px ">
 CSV
</label>
<input class="buttonStyle" type="submit" name="button" value="Generate" style="width:150px;" />
</div>
</div>
</div>
</div>
HtmL.EndForm();
}
`
```

Now, Run and export the report with specified export format in your Report Writer application.

To learn more about export a report with data visualization report items, refer to the [Export data visualization report items](#) section.

Congratulations! You've completed your first ASP.NET Core Writer application!.Click [here](#) to download the already created ASP.NET Core Report Writer application.

## Export RDLC Report

You can export the RDLC reports that exist on the local file system with custom business object data collection. The following steps demonstrates how to export a RDLC report using Report Writer.

This section requires an ASP.NET Core Report Writer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

### Bind data source in Web API controller

The following steps help you to configure the Web API to render the RDLC report with business object data collection.

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
public class ProductList
{
 public string ProductName { get; set; }
 public string OrderId { get; set; }
 public double Price { get; set; }
 public string Category { get; set; }
 public string Ingredients { get; set; }
 public string ProductImage { get; set; }
 public static IList GetData()
 {
 List<ProductList> datas = new List<ProductList>();
 ProductList data = new ProductList()
 {
 ProductName = "Baked Chicken and Cheese",
 OrderId = "323B60",
 Price = 55,
 Category = "Non-Veg",
 Ingredients = "grilled chicken, corn and olives.",
 ProductImage = ""
 };
 datas.Add(data);
 }
}
```

```
data = new ProductList()
{
 ProductName = "Chicken Delite",
 OrderId = "323B61",
 Price = 100,
 Category = "Non-Veg",
 Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
 ProductImage = ""
};

datas.Add(data);

data = new ProductList()
{
 ProductName = "Chicken Tikka",
 OrderId = "323B62",
 Price = 64,
 Category = "Non-Veg",
 Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
 ProductImage = ""
};

datas.Add(data);

return datas;
}

}
`
```

In this tutorial, the `Product List.rdlc` report is used, and it can be downloaded [here](#). Copy and paste the sample RDLC reports into the `wwwroot/Resources` folder. For more information, see [Samples and demos](#).

Create a folder `Resources` into the `wwwroot` folder in your application. Copy and paste the sample RDLC reports into the `Resources` folder.

Open the `HomeController.cs` file in your application and add the `Export()`function to load the report as stream. Refer to the following code snippet.

```
`csharp
```

```

public class HomeController : Controller
{
 // IHostingEnvironment used with sample to get the application data from wwwroot.
 private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
 // IHostingEnvironment initialized with controller to get the data from application data folder.
 public HomeController(Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
 {
 _hostingEnvironment = hostingEnvironment;
 }
 [HttpPost]
 public IActionResult Export(string writerFormat)
 {
 // Here, we have loaded the Product List.rdlc sample report from application the Resources folder.
 FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath +
 @"\Resources\Product List.rdlc", FileMode.Open, FileAccess.Read);

 }
}
```

```

Initialize the Report Writer instance with created reportStream and Set the value of the ReportProcessingMode property value ProcessingMode.Local to provide the dataset collection for that RDLC report.

```

`csharp
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);
writer.ReportProcessingMode = ProcessingMode.Local;
`
```

Bind the business object data values collection by adding a new item to the DataSources as in the following code snippet.

```

`csharp
//Pass the dataset collection for report
writer.DataSources.Clear();
writer.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list", Value =
ProductList.GetData() });
```

Data source **Name** is case sensitive and it should be same as in the data source name in the report definition and the **Value** accepts IList, DataSet, and DataTable inputs.

You can use the **Save** method in Report Writer to generate the export document along with information of the report stream, it will return the generated file as Stream.

```
'csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
    // Here, we have loaded the Product List.rdlc sample report from application the Resources folder.
    FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath +
        @"\Resources\Product List.rdlc", FileMode.Open, FileAccess.Read);
    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);
    writer.ReportProcessingMode = ProcessingMode.Local;
    // Pass the dataset collection for report
    writer.DataSources.Clear();
    writer.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list", Value =
        ProductList.GetData() });
    string fileName = null;
    WriterFormat format;
    string type = null;
    if (writerFormat == "PDF")
    {
        fileName = "Product List.pdf";
        type = "pdf";
        format = WriterFormat.PDF;
    }
    else if (writerFormat == "Word")
    {
        fileName = "Product List.doc";
        type = "doc";
        format = WriterFormat.Word;
    }
}
```

```
else if (writerFormat == "CSV")
{
    fileName = "Product List.csv";
    type = "csv";
    format = WriterFormat.CSV;
}
else
{
    fileName = "Product List.xls";
    type = "xls";
    format = WriterFormat.Excel;
}

MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
`
```

Now, the RDLC report exported successfully in your application.

Export SSRS Report Server Report

Report Writer has support to Export RDL reports into popular file formats such as PDF, Microsoft Word, Microsoft Excel, and CSV from SSRS Report Server.

This section requires an ASP.NET Core Report Writer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

Pure .Net Core project will support only SSRS 2017 server. The ASP.NET Core project with the combination .NET framework will render report from all SSRS server versions.

To export the SSRS Reports, initialize the Report Writer instance and set the `ReportProcessingMode`, `ReportServerURL`, and `ReportPath` properties in Web API as shown in the following code snippet.

```
'csharp
```

```
[HttpPost]
```

```
public IActionResult Export(string writerFormat)
{
    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();
    writer.ReportProcessingMode = ProcessingMode.Remote;
    writer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
    writer.ReportPath = "/Report_Path";
}
```

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of `/folder name/report name`.

[Network credentials for SSRS](#)

The network credentials are required to load specified SSRS report from the specified SSRS Report Server using the Report Writer. Specify the `ReportServerCredential` property in writer instance.

```
'csharp
```

```
writer.ReportServerCredential = new System.Net.NetworkCredential("username", "password");
```

[Set data source credential to shared data sources](#)

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS Server. If the report has any data source that uses credentials to connect with the database, then you should specify the `DataSourceCredentials` for each report data source to establish database connection.

```
'csharp
```

```
List<BoldReports.Web.DataSourceCredentials> dataSourceCredentialsList = new
List<BoldReports.Web.DataSourceCredentials>();

dataSourceCredentialsList.Add(new BoldReports.Web.DataSourceCredentials("datasource name",
"username", "password"));

writer.SetDataSourceCredentials(dataSourceCredentialsList);
```

Data source credentials should be added to the shared data sources that do not have credentials in the connection strings.

Refer to the following final code snippet example to export the SSRS report server reports.

```
'csharp
```

```
[HttpPost]
```

```
public IActionResult Export(string writerFormat)
{
```

```
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();
writer.ReportProcessingMode = ProcessingMode.Remote;
writer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
writer.ReportPath = "/Report_Path";
writer.ReportServerCredential = new System.Net.NetworkCredential("username", "password");
List<BoldReports.Web.DataSourceCredentials> dataSourceCredentialsList = new
List<BoldReports.Web.DataSourceCredentials>();
dataSourceCredentialsList.Add(new BoldReports.Web.DataSourceCredentials("Datasource name",
"username", "password"));
writer.SetDataSourceCredentials(dataSourceCredentialsList);
string fileName = null;
WriterFormat format;
string type = null;
if (writerFormat == "PDF")
{
    fileName = "sales-order-detail.pdf";
    type = "pdf";
    format = WriterFormat.PDF;
}
else if (writerFormat == "Word")
{
    fileName = "sales-order-detail.doc";
    type = "doc";
    format = WriterFormat.Word;
}
else if (writerFormat == "CSV")
{
    fileName = "sales-order-detail.csv";
    type = "csv";
    format = WriterFormat.CSV;
}
else
{
    fileName = "sales-order-detail.xls";
}
```

```

type = "xls";
format = WriterFormat.Excel;
}
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
`
```

Render linked reports

You can export a linked report that points to an existing report, which is published in the SSRS Report Server. You can set the parameter, data source, credential, and other properties like normal SSRS reports using the Report Writer.

```

`csharp
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();
writer.ReportProcessingMode = ProcessingMode.Remote;
writer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
writer.ReportPath = "/LinkedReportPath";
`
```

Export SharePoint Server Report

Report Writer has support to Export RDL reports into popular file formats like PDF, Microsoft Word, Microsoft Excel, and CSV from SharePoint server reports.

This section requires an ASP.NET Core Report Writer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

To export the SharePoint server reports, Initialize the Report Writer and set the `ReportProcessingMode`, `ReportServerURL`, and `ReportPath` properties in Web API as shown in the following code snippet.

```

`csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();
```

```

writer.ReportProcessingMode = ProcessingMode.Remote;
writer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
writer.ReportPath = "http://<servername>/reportserver$instanceName/Report_Path";
}
'

```

In SharePoint server, the **ReportServerUrl** will be same as your site URL. The **ReportPath** is relative to the Report Server URL with the file extension.

[Forms credential for SharePoint Server](#)

The forms credentials are required to load the SharePoint integrated SSRS report from the specified SharePoint integrated SSRS Report Server using the Report Viewer. Specify the **ReportServerFormsCredential** property to access the share point reports.

```

`csharp
writer.ReportServerFormsCredential = new
BoldReports.Web.ReportServerFormsCredential("username", "password");
'

```

[Set data source credential to shared data sources](#)

The shared data source credentials can be added to the **DataSourceCredentials** property to connect with the database.

```

`csharp
List<BoldReports.Web.DataSourceCredentials> dataSourceCredentialsList = new
List<BoldReports.Web.DataSourceCredentials>();

dataSourceCredentialsList.Add(new BoldReports.Web.DataSourceCredentials("datasource name",
"username", "password"));

writer.SetDataSourceCredentials(dataSourceCredentialsList);
'

```

Data source credentials should be added to shared data sources that do not have credentials in the connection strings.

Refer to the following final code snippet example to export the SharePoint server reports.

```

`csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();

    writer.ReportProcessingMode = ProcessingMode.Remote;
    writer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
    writer.ReportPath = "http://<servername>/reportserver$instanceName/Report_Path";
}

```

```
writer.ReportServerFormsCredential = new  
BoldReports.Web.ReportServerFormsCredential("username", "password");  
  
List<BoldReports.Web.DataSourceCredentials> dataSourceCredentialsList = new  
List<BoldReports.Web.DataSourceCredentials>();  
  
dataSourceCredentialsList.Add(new BoldReports.Web.DataSourceCredentials("datasource name",  
"username", "password"));  
  
writer.SetDataSourceCredentials(dataSourceCredentialsList);  
  
string fileName = null;  
  
WriterFormat format;  
  
string type = null;  
  
if (writerFormat == "PDF")  
{  
  
    fileName = "sales-order-detail.pdf";  
    type = "pdf";  
    format = WriterFormat.PDF;  
}  
  
else if (writerFormat == "Word")  
{  
  
    fileName = "sales-order-detail.doc";  
    type = "doc";  
    format = WriterFormat.Word;  
}  
  
else if (writerFormat == "CSV")  
{  
  
    fileName = "sales-order-detail.csv";  
    type = "csv";  
    format = WriterFormat.CSV;  
}  
  
else  
{  
  
    fileName = "sales-order-detail.xls";  
    type = "xls";  
    format = WriterFormat.Excel;  
}
```

```
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
`
```

Export Subreport

You can export a Main report with subreport with popular file formats such as PDF, Microsoft Word, and Microsoft Excel without previewing the report in webpage using ASP.NET Report Writer application.

This section requires an ASP.NET Core Report Writer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

Export RDL Subreport

In this tutorial, the `SideBySideMainReport.rdl`, `SideBySideSubReport.rdl` reports is used, and it can be downloaded [here](#). You can get the report from Bold Reports installation location. The reports used from installed location requires `NorthwindIO_Reports.sdf` database to run, so add the database to your application. For more information, refer to [Samples and demos](#).

Refer to the following steps to export the RDL sub report with specified format.

Create a folder `Resources` into the `wwwroot` folder in your application. Copy and paste the sample RDL reports into the `Resources` folder.

Open the `HomeController.cs` file in your application and add the `Export()`function to load the report as stream. Refer to the following code snippet.

```
'csharp
public class HomeController : Controller
{
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // IHostingEnvironment initialized with controller to get the data from application data folder.
    public HomeController(Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _hostingEnvironment = hostingEnvironment;
    }
}
```

```
[HttpPost]
public IActionResult Export(string writerFormat)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sample reports from application the wwwroot\Resources folder.
    FileStream mainReportStream = new FileStream(basePath + @"\Resources\SideBySideMainReport.rdl",
        FileMode.Open, FileAccess.Read);
    FileStream subReportStream = new FileStream(basePath + @"\Resources\SideBySideSubReport.rdl",
        FileMode.Open, FileAccess.Read);

    .....
}
```

Initialize the Report Writer instance and pass the subreport name and stream to the `LoadSubreport()` method in Report Writer instance.

```
`csharp
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();
writer.LoadSubreport("SideBySideSubReport", subReportStream);`
```

After loading the main report stream using Report Writer instance. Refer to the following code snippet.

```
`csharp
writer.LoadReport(mainReportStream);`
```

You can use the `Save` method in Report Writer to generate the export document along with information of the report stream, it will return the generated file as Stream.

```
`csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sample reports from application the wwwroot\Resources folder;
```

```
FileStream mainReportStream = new FileStream(basePath + @"\Resources\SideBySideMainReport.rdl",
 FileMode.Open, FileAccess.Read);

FileStream subReportStream = new FileStream(basePath + @"\Resources\SideBySideSubReport.rdl",
 FileMode.Open, FileAccess.Read);

BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();

writer.LoadSubreport("SideBySideSubReport", subReportStream);

writer.LoadReport(mainReportStream);

string fileName = null;

WriterFormat format;

string type = null;

if (writerFormat == "PDF")

{

    fileName = "SideBySideMainReport.pdf";

    type = "pdf";

    format = WriterFormat.PDF;

}

else if (writerFormat == "Word")

{

    fileName = "SideBySideMainReport.doc";

    type = "doc";

    format = WriterFormat.Word;

}

else if (writerFormat == "CSV")

{

    fileName = "SideBySideMainReport.csv";

    type = "csv";

    format = WriterFormat.CSV;

}

else

{

    fileName = "SideBySideMainReport.xls";

    type = "xls";

    format = WriterFormat.Excel;

}
```

```
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
`
```

Now, the RDL report exported successfully in your application.

Export RDLC subreport

To export the RDLC report along with subreport, we need to load the subreport streams before loading a main report to the Report Writer as in the following code example, then only subreport processing event will work.

```
`csharp
[HttpPost]
public IActionResult Pdf()
{
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sample report from application the folder wwwroot.
    FileStream inputStream = new FileStream(basePath + @"\Reports\MainReport.rdlc", FileMode.Open,
    FileAccess.Read);
    FileStream subreport1Stream = new FileStream(basePath + @"\Reports\SubReport1.rdlc",
    FileMode.Open, FileAccess.Read);
    FileStream subreport2Stream = new FileStream(basePath + @"\Reports\SubReport2.rdlc",
    FileMode.Open, FileAccess.Read);
    ReportWriter writer = new ReportWriter();
    writer.ReportProcessingMode = ProcessingMode.Local;
    writer.SubreportProcessing += ReportWriter_SubreportProcessing;
    //Adding sub report stream going to be used with exporting report.
    writer.LoadSubreport("SubReport1", subreport1Stream);
    writer.LoadSubreport("SubReport2", subreport2Stream);
    //Loading the report going to export as PDF.
    writer.LoadReport(inputStream);
```

```
writer.DataSources.Clear();
writer.DataSources.Add(new ReportDataSource { Name = "DataSet", Value = MainReport.GetData() });
// Steps to generate PDF report using Report Writer.
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, BoldReports.Writer.WriterFormat.PDF);
// Download the generated from client.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/pdf");
fileStreamResult.FileDownloadName = "Invoice.pdf";
return fileStreamResult;
}
private void ReportWriter_SubreportProcessing(object sender, SubreportProcessingEventArgs e)
{
if (e.ReportPath == "SubReport1")
{
//Pass the dataset collection for subreport
e.DataSources.Clear();
e.DataSources.Add(new ReportDataSource { Name = "DataSet1", Value = SubReport1.GetData() });
}
else if (e.ReportPath == "SubReport2")
{
//Pass the dataset collection for subreport
e.DataSources.Clear();
e.DataSources.Add(new ReportDataSource { Name = "DataSet2", Value = SubReport2.GetData() });
}
}
```

Export Parameter Report

You can set report parameter default values or modify the values using the `SetParameters()` method of Report Writer instance. This section describes how to modify the exported document report parameter values.

This section requires an ASP.NET Core Report Writer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

In this tutorial, the `sales-order-detail.rdl` report is used and it can be downloaded [here](#). You can get the reports from Bold Reports installation location. For more information, refer to [samples and demos](#) section.

Set report parameters to export file

Create a folder `Resources` into the `wwwroot` folder in your application. Copy and paste the sample RDL reports into the `Resources` folder.

To load the report as stream from the application `Resources` folder using the `FileStream` class.

```
'csharp
public class HomeController : Controller
{
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // IHostingEnvironment initialized with controller to get the data from application data folder.
    public HomeController(Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _hostingEnvironment = hostingEnvironment;
    }
    [HttpPost]
    public IActionResult Export(string writerFormat)
    {
        // Here, we have loaded the sample reports from application the wwwroot\Resources folder.
        FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath + @"\Resources\sales-
order-detail.rdl", FileMode.Open, FileAccess.Read);
        BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);
        writer.ReportProcessingMode = ProcessingMode.Remote;
        .....
    }
}
```

You can add collection of report parameters programmatically using the `ReportParameter` property that is the list type of `Report Parameter` class. Refer to the following code snippet to set the report parameter name and value.

```
'csharp
```

```
List<BoldReports.Web.ReportParameter> userParameters = new  
List<BoldReports.Web.ReportParameter>();  
  
userParameters.Add(new BoldReports.Web.ReportParameter()  
{  
  
Name = "SalesOrderNumber",  
  
Values = new List<string>() { "SO50756" }  
});  
  
writer.SetParameters(userParameters);  
'
```

You can use the `Save` method in Report Writer to generate the export document along with information of the report stream, it will return the generated file as Stream.

```
'csharp  
[HttpPost]  
  
public IActionResult Export(string writerFormat)  
{  
  
// Here, we have loaded the sample reports from application the wwwroot\Resources folder.  
FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath + @"\Resources\sales-  
order-detail.rdl", FileMode.Open, FileAccess.Read);  
  
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);  
  
writer.ReportProcessingMode = ProcessingMode.Remote;  
  
List<BoldReports.Web.ReportParameter> userParameters = new  
List<BoldReports.Web.ReportParameter>();  
  
userParameters.Add(new BoldReports.Web.ReportParameter()  
{  
  
Name = "SalesOrderNumber",  
  
Values = new List<string>() { "SO50756" }  
});  
  
writer.SetParameters(userParameters);  
  
string fileName = null;  
  
WriterFormat format;  
  
string type = null;  
  
if (writerFormat == "PDF")  
{  
  
fileName = "sales-order-detail.pdf";
```

```
type = "pdf";
format = WriterFormat.PDF;
}
else if (writerFormat == "Word")
{
fileName = "sales-order-detail.doc";
type = "doc";
format = WriterFormat.Word;
}
else if (writerFormat == "CSV")
{
fileName = "sales-order-detail.csv";
type = "csv";
format = WriterFormat.CSV;
}
else
{
fileName = "sales-order-detail.xls";
type = "xls";
format = WriterFormat.Excel;
}
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
`
```

Now, the parameter report exported successfully in your application.

Report Writer Events

You can handle the Report Writer events with reports using the following events.

SubreportProcessing

ReportErrorOccurred

Subreport Processing

The SubreportProcessing event occurs when subreport is loaded and it is ready to start the processing. You can handle the event and specify the data source, parameters, and subreport loading. The following sample code loads a subreport by assigning the report data source credentials and parameter values in the SubreportProcessing event.

Report Writer instance used to register the SubreportProcessing event in your Report Writer application.

```
'csharp
writer.SubreportProcessing += (sen, arg) =>
{
    // Assign the data source and parameter values to the sub report.
};
```

Set the subreport data source details to the subreport processing event argument.

```
'csharp
writer.SubreportProcessing += (sen, arg) =>
{
    arg.DataSources.Add(new BoldReports.Web.ReportDataSource {Name= "Datasource name", Value =
value });
}
```

Data source Name is case sensitive and it should be same as in the data source name in the report definition. The Value accepts IList, DataSet, and DataTable inputs.

Set the subreport parameters value to the subreport processing event argument.

```
'csharp
writer.SubreportProcessing += (sen, arg) =>
{
```

```
arg.Parameters.Add(new BoldReports.Web.ReportParameterInfo { Name = "Parameter name", Values = values});  
}  
,
```

Report Error

The `ReportErrorOccurred` event raises when an error occurs in the report processing. You can handle the event and get the report error details from the event arguments. Refer to the following sample code to handle the report errors in the `ReportErrorOccurred` event.

```
'csharp  
writer.ReportErrorOccurred += (sen, arg) =>  
{  
    // You can handle the report errors using event arguments.  
};  
,
```

Error logging in ASP.NET Core Report Writer

If an error occurred in report processing, ASP.NET Core Report writer `ReportErrorOccurred` event raised. You can register the event and get the report error details from the event arguments to save all logs, stack trace, and error information into a physical file location.

This section explains how to log the detailed error information to your ASP.NET Core application.

This section requires an ASP.NET Core Report Writer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

In Solution Explorer, open the Report Writer Controller file.

Add the following sample code to register the report errors in the `ReportErrorOccurred` event.

```
'csharp  
[HttpPost]  
public IActionResult Export(string writerFormat)  
{  
    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();  
    writer.ReportErrorOccurred += Writer_ReportErrorOccurred;  
}  
private void Writer_ReportErrorOccurred(object sender, ReportErrorOccurredEventArgs e)  
{  
    // You can register the report errors using event arguments.
```

```
}
```

Create a method in `HomeController` to write the error text into application folder.

```
`csharp
internal void WriteLogs(string errorMessage)
{
    string filePath = Path.Combine(_hostingEnvironment.WebRootPath, "ErrorDetails.txt");
    using (StreamWriter writer = new StreamWriter(filePath, true))
    {
        writer.AutoFlush = true;
        writer.WriteLine(errorMessage);
    }
}
```

Invoke the newly created function in `ReportErrorOccurred` as follows.

```
`csharp
private void Writer_ReportErrorOccurred(object sender, ReportErrorOccurredEventArgs e)
{
    WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2}", e.ClassName,
    e.MethodName, e.Message));
}
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final controller is given as follows, you can replace it in your application.

```
`csharp
[HttpPost]
public IActionResult Export(string writerFormat)
{
    // Here, we have loaded the sales-order-detail sample report from application the folder
    wwwroot\Resources.
```

```
FileStream reportStream = new FileStream(_hostingEnvironment.WebRootPath + @"\Resources\sales-order-detail.rdl", FileMode.Open, FileAccess.Read);

BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);

writer.ReportErrorOccurred += Writer_ReportErrorOccurred;

string fileName = null;

WriterFormat format;

string type = null;

if (writerFormat == "PDF")

{

    fileName = "sales-order-detail.pdf";

    type = "pdf";

    format = WriterFormat.PDF;

}

else if (writerFormat == "Word")

{

    fileName = "sales-order-detail.doc";

    type = "doc";

    format = WriterFormat.Word;

}

else if (writerFormat == "CSV")

{

    fileName = "sales-order-detail.csv";

    type = "csv";

    format = WriterFormat.CSV;

}

else

{

    fileName = "sales-order-detail.xls";

    type = "xls";

    format = WriterFormat.Excel;

}

MemoryStream memoryStream = new MemoryStream();

writer.Save(memoryStream, format);
```

```
// Download the generated export document to the client side.  
memoryStream.Position = 0;  
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);  
fileStreamResult.FileDownloadName = fileName;  
return fileStreamResult;  
}  
  
private void Writer_ReportErrorOccurred(object sender, ReportErrorOccurredEventArgs e)  
{  
    WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2}", e.ClassName,  
    e.MethodName, e.Message));  
}  
  
internal void WriteLogs(string errorMessage)  
{  
    string filePath = Path.Combine(_hostingEnvironment.WebRootPath, "ErrorDetails.txt");  
    using (StreamWriter writer = new StreamWriter(filePath, true))  
    {  
        writer.AutoFlush = true;  
        writer.WriteLine(errorMessage);  
    }  
}
```

Encrypt and Secure Documents

Encrypt and Secure Documents option allows you to protect the exported document such as PDF, Word, and Excel from unauthorized users by encrypting the document using the encryption password. The following code snippet explains how to encrypt the exported document with the user defined password.

Password Protected PDF document

You can protect the exported PDF document using the following code snippet.

```
'csharp  
writer.PDFOptions = new BoldReports.Writer.PDFOptions();  
writer.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity  
{  
    UserPassword = "Password"  
};  
'
```

Password Protected Word document

You can protect the exported Word document using the following code snippet.

```
'csharp  
writer.WordOptions = new BoldReports.Writer.WordOptions()  
{  
    EncryptionPassword = "password"  
};  
'
```

Password Protected Excel document

You can protect the exported Excel document using the following code snippet.

```
'csharp  
writer.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    PasswordToModify = "password",  
    PasswordToOpen = "password"  
};  
'
```

Advance Layouts For Merged Cells

Report Writer supports additional export layouts in Word and Excel export formats to eliminate the tiny columns, rows, and merged cells. The benefits of the layout are:

Overcomes the merging problem with tiny cells, rows, and columns in SSRS Excel and Word exporting.

Provides clear readability by eliminating the tiny columns, rows, and merge cells.

Allows you to perform data manipulations such as applying sorting, filters, and grouping in Excel.

Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the `LayoutOption` to `TopLevel`. The `ParagraphSpacing` is the distance value added between two elements in the document.

```
'csharp  
writer.WordOptions.LayoutOption = WordLayoutOptions.TopLevel;  
writer.WordOptions.ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()  
{  
    Bottom = 0.5f,  
'
```

```
Top = 0.5f  
};  
`
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior. The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, add an empty paragraph between two tables.

[Excel document advance layout for merged cells](#)

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the `LayoutOption` as `IgnoreCellMerge`.

```
`csharp  
writer.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge  
};  
`
```

[Change the Default File Format](#)

Users can change the default file format to any other file format that is supported by the selected file type. It includes APIs to set the formats before exporting the document. Refer to the following section to change the default file formats.

[Change the Word document type](#)

You can save the report to the required Word document version by setting the `FormatType` property.

```
`csharp  
writer.WordOptions = new BoldReports.Writer.WordOptions()  
{  
    FormatType = WordFormatType.Docx  
};  
`
```

[Change the Excel document type](#)

You can save the report to the required Excel document version by setting the `ExcelSaveType` property.

```
`csharp  
writer.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013  
};
```

Change the CSV document type

You can save the report to the required CSV file extension by setting the `FileExtension` property.

```
'csharp  
writer.CsvOptions = new BoldReports.Writer.CsvOptions()  
{  
    FileExtension = ".txt"  
};  
'
```

PDF Settings

The PDF export options provides properties to manage PDF export behaviors. You can set the customization properties in the `PDFOptions`. Refer to the following code snippet to initialize the `PDFOptions` property.

```
'csharp  
writer.PDFOptions = new PDFOptions();  
'
```

Export with complex scripts

To export reports with the complex script language texts, set the `ComplexScript` property of `PDFOptions` instance to true.

```
'csharp  
writer.PDFOptions.EnableComplexScript = true;  
'
```

PDF conformance

You can export the report as a PDF/A-1b document by specifying the `PdfConformanceLevel.Pdf_A1B` conformance level in the `PdfConformanceLevel` property.

```
'csharp  
writer.PDFOptions.PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B;  
'
```

Embedding Custom PDF fonts

You can add custom fonts to the PDF exported document by adding the font streams to `Fonts` collection in `PDFOptions` instance.

To add custom fonts to the PDF exported document, follow these steps:

Add the font .ttf files to your application `wwwroot/Resources` folder.

In the Solution Explorer, open the properties of the font file and set the **Copy** property to Output Directory as Copy always.

Initialize the **Font** collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp
//Load Missing font stream to the pdf document
writer.PDFOptions.Fonts = new Dictionary<string, System.IO.Stream>
{
    { "Segoe UI", new FileStream(basePath + @"\Resources\Fontfilename.ttf", FileMode.Open,
        FileAccess.Read) }
};
```

If any fonts used in the report definition is not installed or available in the local system, then you should load the font stream like above code snippet.

[Password Protected PDF document](#)

Allows you to protect the exported PDF document from unauthorized users by encrypting the document using encryption password. The following code snippet explains how to encrypt the exported document with user-defined password.

```
'csharp
writer.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity
{
    UserPassword = "Password"
};
```

[Excel Settings](#)

The Excel export options provides properties to manage Excel document export behaviors. You can set the customization properties in the **ExcelOptions**. Refer to the following code snippet to initialize the **ExcelOptions** property.

```
'csharp
writer.ExcelOptions = new ExcelOptions();
`
```

[Excel document type](#)

You can save the report to the required Excel version by setting the **ExcelSaveType** property.

```
'csharp
```

```
writer.ExcelOptions.ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013;
```

Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the `LayoutOption` as `IgnoreCellMerge`.

```
'csharp
```

```
writer.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge  
};
```

Protecting Excel document from editing

You can restrict the Excel document from editing by providing the `ExcelSheetProtection` or enabling the `ReadOnlyRecommended` properties.

```
'csharp
```

```
writer.ExcelOptions.ReadOnlyRecommended = true;  
writer.ExcelOptions.ExcelSheetProtection = Syncfusion.XlsIO.ExcelSheetProtection.DeletingColumns;
```

Change Excel export format

Allows you to change the default file format to any other file format using the `ExcelSaveType` properties.

```
'csharp
```

```
writer.ExcelOptions.ExcelSaveType = ExcelVersion.Excel2013;
```

Password Protected Excel document

Allows you to protect the exported Excel document from unauthorized users by encrypting the document using the encryption password. The following code snippet explains how to encrypt the exported document with user-defined password.

```
'csharp
```

```
writer.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
    PasswordToModify = "password",  
    PasswordToOpen = "password"  
};
```

Word Settings

The Word export options provides properties to manage Word document export behaviors. You can set the customization properties in the `WordOptions`. Refer to the following code snippet to initialize the `WordOptions` property.

```
'csharp  
writer.WordOptions = new WordOptions();  
'
```

Word document type

You can save the report to the required document version by setting the `FormatType` property.

```
'csharp  
writer.WordOptions.FormatType = WordFormatType.Docx;  
'
```

Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the `LayoutOption` to `TopLevel`. The `ParagraphSpacing` is the distance value added between two elements in the document.

```
'csharp  
writer.WordOptions.LayoutOption = WordLayoutOptions.TopLevel;  
writer.WordOptions.ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()  
{  
    Bottom = 0.5f,  
    Top = 0.5f  
};  
'
```

A paragraph element is inserted between two tables in the exported document to overcome the Word document auto merging behavior. The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, add an empty paragraph between two tables.

Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without password. The following are the types of protection:

`AllowOnlyComments`: Adds or modifies only the comments in the Word document.

`AllowOnlyFormFields`: Modifies the form field values in the Word document.

`AllowOnlyRevisions`: Accepts or rejects the revisions in the Word document.

AllowOnlyReading: Views the content only in the Word document.

NoProtection: Accesses or edits the Word document contents as normally.

'csharp

```
writer.WordOptions.ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading;
```

'

Password Protected Word document

Allows you protect the exported Word document from unauthorized users by encrypting the document using encryption password. The following code snippet explains how to encrypt the exported document with user-defined password.

'csharp

```
writer.WordOptions = new BoldReports.Writer.WordOptions()
```

{

```
EncryptionPassword = "password"
```

};

'

CSV Settings

The **CsvOptions** allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document. You should set the customization properties in the **CsvOptions** property of Report Writer instance.

'csharp

```
writer.CsvOptions = writer.CsvOptions = new BoldReports.Writer.CsvOptions()
```

{

```
Encoding = System.Text.Encoding.Default,
```

```
FieldDelimiter = ",",
```

```
UseFormattedValues = false,
```

```
Qualifier = "#",
```

```
RecordDelimiter = "@",
```

```
SuppressLineBreaks = true,
```

```
FileExtension = ".txt"
```

};

'

Export Data Visualization in Core Environment

Report Writer uses **WebBrowser** to export the data visualization to PDF, Word, Excel, and CSV file formats. The **WebBrowser** is not supported in Core environment. To overcome this limitation in Core environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

It supports **ASP.NET Core 2.1 or above** version only.

To download **PhantomJS** application and deploy it on your machine, you should accept its license terms on [LICENSE](#) and [Third-Party](#) document.

Download **PhantomJS** [here](#) and extract the download file.

Copy the **PhantomJS.exe** file from the extracted bin folder and paste into **wwwroot/PhantomJS** folder in your application.

Open the Report Writer application Web API file.

Set the **UsePhantomJS** property to **true**.

Set the **PhantomJSPath**, **Scripts** and **DependentScripts** property in Report Writer instance. The following code example demonstrates how to configure **PhantomJS** in your ASP.NET Core application.

In this tutorial, the **product-line-sales.rdl** report is used, and it can be downloaded in this [link](#).

```
'csharp
public class HomeController : Controller
{
    // IHostingEnvironment used with sample to get the application data from wwwroot.
    private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
    // IHostingEnvironment initialized with controller to get the data from application data folder.
    public HomeController(Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
    {
        _hostingEnvironment = hostingEnvironment;
    }
    public IActionResult Index()
    {
        return View();
    }
    [HttpPost]
```

```
public IActionResult Export(string writerFormat)
{
    string fileName = null;
    WriterFormat format;
    string basePath = _hostingEnvironment.WebRootPath;
    // Here, we have loaded the sample report from application the folder wwwroot.
    FileStream inputStream = new FileStream(basePath + @"\Resources\product-line-sales.rdl",
    FileMode.Open, FileAccess.Read);
    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(inputStream);
    // PhantomJS Option to export the data visualization report items.
    writer.ExportResources.UsePhantomJS = true;
    writer.ExportResources.PhantomJSPath = basePath + @"\PhantomJS\";
    writer.ExportResources.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
        //Gauge component scripts
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
        //Map component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
        //Report Viewer Script
        "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
    };
    writer.ExportResources.DependentScripts = new List<string>
    {
        "https://code.jquery.com/jquery-1.10.2.min.js"
    }
}
```

```
};

if (writerFormat == "PDF")
{
    fileName = "ProductLineSales.pdf";
    format = WriterFormat.PDF;
}

else if (writerFormat == "Word")
{
    fileName = "ProductLineSales.doc";
    format = WriterFormat.Word;
}

else if (writerFormat == "CSV")
{
    fileName = "ProductLineSales.CSV";
    format = WriterFormat.CSV;
}

else
{
    fileName = "ProductLineSales.xls";
    format = WriterFormat.Excel;
}

MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated document from client.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/pdf");
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}
}
```

The **Scripts** and **Dependent scripts** must be added to export the data visualization report items.

Limitations

Linux

Report Writer uses **PhantomJS.exe** for data visualization report items rendering, which is not supported with Linux environment. So, visualization report items will not be available in Report Writer generated document.

Custom code

Custom code in a report allows to include new custom constants, variables, functions, or subroutines. As the VBCodeProvider is not available in .NET Core platform, report with custom codes requires additional settings to enable this feature. You can make use of the following solution using custom codes with Report Writer.

[How to use custom code with Report Writer](#)

Data source

For RDL, built-in data source processing support is available only for SQL data source. For other datasources, you have to create a custom data extension for data processing.

PDF fonts

.NET Core platform does not have a support to get font details from system environment to embed report fonts with PDF document. So, Report Writer will generate PDF documents only with TimesNewRoman font. You have to provide report font details as separate with PDF Options to get the PDF documents with expected font used in Report. Refer [embedding-custom-pdf-fonts](#) to resolve this problem.

SSRS support

For SQL Server Reporting Service (SSRS), as of now SQL Server 2017 Reporting Service is only supported to generate the document from Reporting Server.

Excel export

Microsoft Excel has a limitation that you could not create a Excel document with more than 1,048,576 rows and 16,384 columns. So, as per limitation Report Writer does not support to create the Excel document from report above of 1,048,576 rows and 16,384 columns.

How to section for Report Writer component

[how to use custom code with Report Writer](#)

[Generate and export RDL and RDLC reports programmatically](#)

How to use custom code with Report Writer

Custom code in a report allows to include new custom constants, variables, functions, or subroutines. As the VBCodeProvider is not available in .NET Core platform, report with custom codes requires additional settings to enable this feature. This section describes the steps required to use custom codes with Report Writer.

Create a custom code class file

Create a new C# class file in your application.

The namespace should be **BoldReports.Processing.Helper** and class declaration should be **public static class Code**.

Write the C# methods equivalent to your report VB codes.

Here is the example code to convert the value to USD.

```
'csharp
namespace BoldReports.Processing.Helper
{
    public static class Code
    {
        public static string PrintHelloWorld()
        {
            return "Hello World";
        }
    }
}
```

Create a assembly reference and add it to Report Writer

If you have created the custom code class, then you need to add the assemblies in the list with what you have used in your application to Report Writer as shown in following code example.

```
'csharp
using (var reportWriter = new ReportWriter())
{
    reportWriter.Assemblies.Add(this.GetType().GetTypeInfo().Assembly);
    reportWriter.LoadReport(inputStream);
}
```

If you want to use the custom code feature, then you should not use the **ReportWriter(Stream ReportStream)** constructor to load the report and assemblies added with **Assemblies** property before using **LoadReport** method.

How to generate and export RDL and RDLC reports programmatically with ASP.NET Core

The Bold Reports reporting library allows you to generate and export RDL and RDLC reports programmatically with ASP.NET Core. The `ReportDefinition` class is defined as an object model of a report. You can create, add, and modify the report properties, report sections like header and footer and report items in the report definition instance.

The following steps illustrates how to create a basic report object model:

Initialize a report definition

The following code snippet guides you in initializing the report definition.

```
'csharp
```

```
ReportDefinition CreateReport()
{
    ReportDefinition report = new ReportDefinition();
    report.ReportSections = new ReportSections();
    var reportSection = new ReportSection();
    report.ReportSections.Add(reportSection);
    reportSection.Width = new BoldReports.RDL.DOM.Size("6in");
    report.ReportUnitType = "Inch";
    report.RDLType = RDLType.RDL2010;
    return report;
}

BoldReports.RDL.DOM.Style AddStyle()
{
    BoldReports.RDL.DOM.Style style = new BoldReports.RDL.DOM.Style();
    style.Border = new BoldReports.RDL.DOM.Border();
    style.Border.Width = new BoldReports.RDL.DOM.Size("1pt");
    style.Border.Style = "Solid";
    style.Border.Color = "Black";
    return style;
}
```

Create a report page header

The following code snippet guides you in creating a `PageHeader` report section and setting values to their properties.

```
'csharp
```

How to generate and export RDL and RDLC reports programmatically with ASP.NET Core [Create a report page footer](#)

```
ReportDefinition CreateReport()
{
...
...
...
reportSection.Page = new BoldReports.RDL.DOM.Page();
reportSection.Page.Style = AddStyle();
reportSection.Page.PageHeight = "4in";
reportSection.Page.PageWidth = "6in";
reportSection.Page.PageHeader = CreateHeader();
...
...
...
}
PageHeader CreateHeader()
{
    PageHeader pageHeader = new PageHeader();
    pageHeader.Height = new BoldReports.RDL.DOM.Size("0.59167in");
    pageHeader.Style = AddStyle();
    pageHeader.PrintOnFirstPage = true;
    pageHeader.PrintOnLastPage = true;
    pageHeader.ReportItems = new ReportItems();
    var textBox = CreateTextBox("TextBox2", "Header", "1.61333in", "0.13833in", "2.2in", "0.34167in");
    pageHeader.ReportItems.Add(textBox);
    return pageHeader;
}
`
```

[Create a report page footer](#)

The following code snippet guides you in creating a `PageFooter` report section and setting values to their properties.

```
`csharp
ReportDefinition CreateReport()
{
...
`
```

```
...
...
reportSection.Page.PageHeader = CreateFooter();
...
...
...
}
PageFooter CreateFooter()
{
    PageFooter pageFooter = new PageFooter();
    pageFooter.Style = AddStyle();
    pageFooter.Height = new BoldReports.RDL.DOM.Size("0.59167in");
    pageFooter.ReportItems = new ReportItems();
    pageFooter.PrintOnFirstPage = true;
    pageFooter.PrintOnLastPage = true;
    var textBox = CreateTextBox("TextBox3", "Footer", "1.61333in", "0.05416in", "2.2in", "0.34167in");
    pageFooter.ReportItems.Add(textBox);
    return pageFooter;
}
`
```

Initialize a report body section

Initialize a report body object and set the values to their properties like height, styles, and report items as shown in the following code snippet.

```
`csharp
ReportDefinition CreateReport()
{
...
...
...
reportSection.Body = CreateBody();
...
...
...
}
```

```
}

Body CreateBody()
{
    var body = new Body();
    body.Height = new BoldReports.RDL.DOM.Size("2.03333in");
    body.Style = AddStyle();
    body.ReportItems = new ReportItems();
    var textBox = CreateTextBox("TextBox1", "Body", "1.58833in", "0.66333in", "2.225in", "0.59167in");
    body.ReportItems.Add(textBox);
    return body;
}
`
```

Using the following code snippets, you can create a **Textbox** report item and assign the values for their properties like name, styles, paragraphs, and dimension values.

```
`csharp
PageHeader CreateHeader()
{
    ...
    ...
    ...
    ...
    pageHeader.ReportItems = new ReportItems();
    var textBox = CreateTextBox("TextBox2", "Header", "1.61333in", "0.13833in", "2.2in", "0.34167in");
    pageHeader.ReportItems.Add(textBox);
    ...
    ...
    ...
}
PageFooter CreateFooter()
{
    ...
    ...
    ...
    ...
    pageFooter.ReportItems = new ReportItems();
```

How to generate and export RDL and RDLC reports programmatically with ASP.NET Core Initialize a report body section

```
var textBox = CreateTextBox("TextBox3", "Footer", "1.61333in", "0.05416in", "2.2in", "0.34167in");
pageFooter.ReportItems.Add(textBox);

...
...
...
}

Body CreateBody()
{
...
...
...
body.ReportItems = new ReportItems();

var textBox = CreateTextBox("TextBox1", "Body", "1.58833in", "0.66333in", "2.225in", "0.59167in");
body.ReportItems.Add(textBox);

...
...
...
}

BoldReports.RDL.DOM.TextBox CreateTextBox(string name, string text, string left, string top, string width, string height)
{
    var textBox = new BoldReports.RDL.DOM.TextBox();
    textBox.Name = name;
    textBox.Height = new BoldReports.RDL.DOM.Size(height);
    textBox.Width = new BoldReports.RDL.DOM.Size(width);
    textBox.Left = new BoldReports.RDL.DOM.Size(left);
    textBox.Top = new BoldReports.RDL.DOM.Size(top);
    textBox.Style = new BoldReports.RDL.DOM.Style();
    textBox.Style.TextAlign = "Center";
    textBox.Style.VerticalAlign = "Top";
    textBox.Paragraphs = new Paragraphs();
    BoldReports.RDL.DOM.Paragraph paragraph = new BoldReports.RDL.DOM.Paragraph();
    TextRuns runs = new TextRuns();
```

```
TextRun run = new TextRun();
run.Style = new BoldReports.RDL.DOM.Style();
run.Style.FontStyle = "Default";
run.Style.TextAlign = "Center";
run.Style.FontFamily = "Arial";
run.Style.FontSize = new BoldReports.RDL.DOM.Size("10pt");
run.Value = text;
runs.Add(run);
paragraph.Style = new BoldReports.RDL.DOM.Style();
paragraph.Style.VerticalAlign = "Top";
paragraph.Style.TextAlign = "Center";
paragraph.TextRuns = runs;
textBox.Paragraphs.Add(paragraph);
return textBox;
}
```

[Export the generated report in ReportWriter](#)

You can export the report using the ReportWriter, after the report definition or stream is created. The following code snippet illustrates how to export the report into PDF format using the Report Writer by report definition.

```
'csharp
ReportWriter reportWriter = new ReportWriter();
reportWriter.LoadReport(reportDefinition);
reportWriter.Save(fileName, WriterFormat.PDF);
'
```

[Reporting tools for ASP.NET Core](#)

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

[Key features](#)

Data Sources --- Supports connection to major data providers such as

Microsoft SQL Server, Oracle, OLEDB and **ODBC** for exploring data and design reports with a wide range of data sources.

User friendly Environment --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

Report items --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

Report Parameter --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

Expression --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

Add Web Report Designer to an ASP.NET Core application

This section explains the steps required to add Web Report Designer to an ASP.NET Core application. To create an ASP.NET Core application with Bold Report Designer, follow these steps:

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Designer API creation.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

![Project template](/static/assets/aspnet-core/report-designer/images/getting-started/Getting-Started_img1.png)

Choose the **ASP.NET Core version** and select the Web Application **Model-View-Controller** template and then click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/aspnet-core/report-designer/images/getting-started/aspnet-core-web-application-template.png)

If you need to use Bold Reports with ASP.NET Core on Linux or macOS, then refer to this [Can Bold Reports be used with ASP.NET Core on Linux and macOS](#) section.

List of dependency libraries

The Web API service configuration requires the following report server-side packages.

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command and then search the following BoldReports.AspNet.Core and BoldReports.Net.Core packages to install them in the application. The following table provides details about the packages and their usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core package**.

Newtonsoft.Json | Serialize and deserialize the data or report for web report designer. It is a mandatory package for the web report designer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer and designer. It should be referred in project when process the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0 .

Refer Scripts and CSS

Directly refer the scripts and style sheets that are mandatorily required to render the web Report Designer, from **CDN** links.

Open the **\Views\Shared_Layout.cshtml** page and refer the scripts and styles as shown in the below example code.

NOTE : Include the scripts and CSS references under the appropriate environment. (For eg: If your environment is “Development”, then refer the scripts and CSS files under the tag environment `include="Development"`). Refer all the required external and internal scripts only once in the page with proper order.

```
'html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>@ ViewData["Title"] - ReportDesignerSample</title>
<environment include="Development">
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css"
rel="stylesheet" />
<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css"
rel="stylesheet" />
<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css"
rel="stylesheet" />
<!--Render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-
circulargauge.min.js"></script>
</environment>
</head>
<body>
<environment include="Development">
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
<script src="https://cdn.boldreports.com/external/jsrender.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js"
type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js"
type="text/javascript"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-
widgets.min.js"></script>
<!--Chart component script added before the Report Designer and viewer script to render report with
chart report item-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"
type="text/javascript"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js"
type="text/javascript"></script>
</environment>
```

```
@RenderSection("Scripts", required: false)  
</body>  
</html>  
'
```

Refer to the [Dependencies](#) to learn more details about web Report Designer dependent scripts and style sheets links.

Configure Script Manager

Open the the `~/Views/Shared/_Layout.cshtml` page and add the Script Manager at the end of `<body>` element as in the following code sample.

```
'html  
<body>  
....  
....  
<!-- Bold Reporting ScriptManager -->  
<bold-script-manager></bold-script-manager>  
</body>  
'
```

Tag helper

It is necessary to define the following tag helper within the `_ViewImports.cshtml` page to initialize the web Report Designer component with the tag helper support.

```
'js  
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers  
@using BoldReports.TagHelpers  
@addTagHelper *, BoldReports.AspNetCore  
'
```

Initialize Report Designer

Initialize the web Report Designer as shown in the following code sample in any web page (`cshtml`) of your application in the `~/Views` folder. For an example, the `Index.cshtml` page can be replaced with the following code by removing the existing codes

```
'html  
<div style="height: 500px; width: 100%;">  
<bold-report-designer id="designer"></bold-report-designer>  
</div>  
'
```

Add API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportingAPIController**, and then click **Add**.

![Adding a new controller to the project](/static/assets/aspnet-core/report-designer/images/getting-started/add-core-api-controller.png)

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Configure Web API

The **IReportDesignerController** interface contains the required actions and helper methods declaration to process the designer file and data actions. The **ReportDesignerHelper** and **ReportHelper** class contains methods that help to process Post or Get request from the control and return the response.

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the web Report Designer control and returns the response to the web Report Designer control. It has the following methods.

| Methods | Description |
|----------------------------------------------------------------------|-------------|
| ----- ----- | |
| GetResource Returns the report resource for the requested key. | |
| ProcessReport Processes the report request and returns the result. | |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request for report preview action and returns the response to the web Report Designer. It has the following methods.

| Methods | Description |
|----------------------------------------------------------------------|-------------|
| ----- ----- | |
| GetResource Returns the report resource for the requested key. | |
| ProcessReport Processes the report request and returns the result. | |

Open **ReportingAPIController.cs** file and replace the following code example.

```
'csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
```

```
using Microsoft.AspNetCore.Mvc;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
namespace ReportDesignerSample
{
    [Route("api/[controller]/[action]")]
    public class ReportingAPIController : Controller,
        BoldReports.Web.ReportDesigner.IReportDesignerController
    {
        private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
        private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
        public ReportingAPIController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
            Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
        {
            _cache = memoryCache;
            _hostingEnvironment = hostingEnvironment;
        }
        private string GetFilePath(string itemName, string key)
        {
            string targetFolder = this._hostingEnvironment.WebRootPath + "\\";
            targetFolder += "Cache";
            if (!System.IO.Directory.Exists(targetFolder))
            {
                System.IO.Directory.CreateDirectory(targetFolder);
            }
            if (!System.IO.Directory.Exists(targetFolder + "\\\" + key))
            {
                System.IO.Directory.CreateDirectory(targetFolder + "\\\" + key);
            }
            return targetFolder + "\\\" + key + "\\\" + itemName;
        }
        public object GetImage(string key, string image)
        {
            return ReportDesignerHelper.GetImage(key, image, this);
        }
    }
}
```

```
}

public object GetResource(ReportResource resource)
{
    return ReportHelper.GetResource(resource, this, _cache);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //You can update report options here
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

[HttpPost]
public object PostDesignerAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null, this._cache);
}

public object PostFormDesignerAction()
{
    return ReportDesignerHelper.ProcessDesigner(null, this, null, this._cache);
}

public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, this._cache);
}

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
```

```
errorMessage = string.Empty;
if (itemData.Data != null) {
    System.IO.File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
}
else if (itemData.PostedFile != null) {
    var fileName = itemId;
    if (string.IsNullOrEmpty(itemId)) {
        fileName = System.IO.Path.GetFileName(itemData.PostedFile.FileName);
    }
    using(System.IO.MemoryStream stream = new System.IO.MemoryStream())
    {
        itemData.PostedFile.OpenReadStream().CopyTo(stream);
        byte[] bytes = stream.ToArray();
        var writePath = this.GetFilePath(fileName, key);
        System.IO.File.WriteAllBytes(writePath, bytes);
        stream.Close();
        stream.Dispose();
    }
}
return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
    return resource;
}

[HttpPost]
public void UploadReportAction()
{
    ReportDesignerHelper.ProcessDesigner(null, this, this.Request.Form.Files[0], this._cache);
}
```

```
}
```

Set the service URL

To browse, open and save the reports in the application, set the WebAPI controller name to the **ServiceUrl** property of the web Report Designer. You can replace the following code in your web Report Designer page.

```
'html
<div style="height: 500px; width: 100%;">
<bold-report-designer id="designer" service-url="/api/ReportingAPI" ></bold-report-designer>
</div>
'
```

Run the Application

Run the sample application and you can see the web ReportDesigner on the page as displayed in the following screenshot.

![Report Designer demo](/static/assets/aspnet-core/report-designer/images/getting-started/Getting-Started_img9.png)

Add Web Report Designer to an ASP.NET Core Razor Pages

This section explains the steps required to add Web Report Designer to an ASP.NET Core Razor Pages application. To create an ASP.NET Core Razor Pages application with Bold Report Designer, follow these steps:

Bold Reports ASP.NET Core supports from **.NET Core 2.1** only. So, choose the .NET Core version **ASP.NET Core 2.1** or higher versions for Designer API creation.

The source code for this ASP.NET Core Razor Pages application is available on [GitHub](#).

Prerequisites

Before getting started with bold web report designer, make sure your development environment includes the following requirements.

[Visual Studio 2019](#) with ASP.NET and Web Development workloads.

[.NET Core 3.1](#) Framework.

Create ASP.NET Core application

Start Visual Studio 2019 and click **Create new project**.

Choose **ASP.NET Core Web Application**, and then click **Next**.

Change the project name, and then click **Create**.

In the dropdown with the **ASP.NET Core version**, choose **ASP.NET Core 3.1**.

Select the **Web Application** template, then click **Create**.

![Creating a new ASP.NET Core Razor Pages Application Project](/static/assets/aspnet-core/common/aspnet-core-razor-application-template.png)

List of dependency libraries

The Web API service configuration requires the following report server-side packages.

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command and then search the following **BoldReports.AspNetCore** and **BoldReports.NetCore** packages to install them in the application. The following table provides details about the packages and their usage.

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core package**.

Newtonsoft.Json | Serialize and deserialize the data or report for web report designer. It is a mandatory package for the web report designer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer and designer. It should be referred in project when process the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0 .

Refer Scripts and CSS

Directly refer the scripts and style sheets that are mandatorily required to render the web Report Designer, from **CDN** links.

Open the **\Pages\Shared_Layout.cshtml** page and refer the scripts and styles as shown in the below example code.

Replace the following code in your **\Pages\Shared_Layout.cshtml** page tag.

```
'html
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css"
rel="stylesheet" />
```

```
<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css" rel="stylesheet" />
<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css" rel="stylesheet" />
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js"></script>
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js" type="text/javascript"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js" type="text/javascript"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>
<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-widgets.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js"></script>
<!-- Report component script-->
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"></script>
<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js" type="text/javascript"></script>
`
```

Refer to the [Dependencies](#) to learn more details about web Report Designer dependent scripts and style sheets links.

Tag helper

It is necessary to define the following tag helper within the `_ViewImports.cshtml` page to initialize the web Report Designer component with the tag helper support.

```
'js  
@using BoldReports.TagHelpers  
@addTagHelper *, BoldReports.AspNetCore  
'
```

Configure Script Manager

Open the `~/Pages/Shared/_Layout.cshtml` page and add the reporting Script Manager at the end of `<body>` element as in the following code sample.

```
'html  
<body>  
<div>  
    @RenderBody()  
</div>  
    @RenderSection("Scripts", required: false)  
    <!-- Bold Reports script manager -->  
    <bold-script-manager></bold-script-manager>  
</body>  
'
```

Initialize Report Designer

Open the `Index.cshtml` page.

Remove the existing codes and add the following code.

```
'html  
<div style="height: 500px; width: 100%;">  
    <bold-report-designer id="designer"></bold-report-designer>  
</div>  
'
```

Add API Controller

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportingAPIController**, and then click **Add**.

![Adding a new controller to the project](/static/assets/aspnet-core/report-designer/images/getting-started/add-core-api-controller.png)

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Configure Web API

The **IReportDesignerController** interface contains the required actions and helper methods declaration to process the designer file and data actions. The **ReportDesignerHelper** and **ReportHelper** class contains methods that help to process Post or Get request from the control and return the response.

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the web Report Designer control and returns the response to the web Report Designer control. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request for report preview action and returns the response to the web Report Designer. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

Open **ReportingAPIController.cs** file and replace the following code example.

```
'csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
```

```
namespace ReportDesignerSample
{
    [Route("api/[controller]/[action]")]
    public class ReportingAPIController : Controller,
        BoldReports.Web.ReportDesigner.IReportDesignerController
    {
        private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
        private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
        public ReportingAPIController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
            Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
        {
            _cache = memoryCache;
            _hostingEnvironment = hostingEnvironment;
        }
        private string GetFilePath(string itemName, string key)
        {
            string targetFolder = this._hostingEnvironment.WebRootPath + "\\";
            targetFolder += "Cache";
            if (!System.IO.Directory.Exists(targetFolder))
            {
                System.IO.Directory.CreateDirectory(targetFolder);
            }
            if (!System.IO.Directory.Exists(targetFolder + "\\"))
            {
                System.IO.Directory.CreateDirectory(targetFolder + "\\");
            }
            return targetFolder + "\\\" + key + "\\\" + itemName;
        }
        public object GetImage(string key, string image)
        {
            return ReportDesignerHelper.GetImage(key, image, this);
        }
        public object GetResource(ReportResource resource)
        {
```

```
return ReportHelper.GetResource(resource, this, _cache);
}

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //You can update report options here
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    //You can update report options here
}

[HttpPost]
public object PostDesignerAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportDesignerHelper.ProcessDesigner(jsonResult, this, null, this._cache);
}

public object PostFormDesignerAction()
{
    return ReportDesignerHelper.ProcessDesigner(null, this, null, this._cache);
}

public object PostFormReportAction()
{
    return ReportHelper.ProcessReport(null, this, this._cache);
}

[HttpPost]
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    return ReportHelper.ProcessReport(jsonResult, this, this._cache);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null) {
        System.IO.File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
}
```

```
}

else if (itemData.PostedFile != null) {

var fileName = itemId;
if (string.IsNullOrEmpty(itemId)) {
fileName = System.IO.Path.GetFileName(itemData.PostedFile.FileName);
}

using(System.IO.MemoryStream stream = new System.IO.MemoryStream())
{
itemData.PostedFile.OpenReadStream().CopyTo(stream);
byte[] bytes = stream.ToArray();
var writePath = this.GetFilePath(fileName, key);
System.IO.File.WriteAllBytes(writePath, bytes);
stream.Close();
stream.Dispose();
}
}

return true;
}

public ResourceInfo GetData(string key, string itemId)
{
var resource = new ResourceInfo();
resource.Data = System.IO.File.ReadAllBytes(this.GetFilePath(itemId, key));
return resource;
}

[HttpPost]
public void UploadReportAction()
{
ReportDesignerHelper.ProcessDesigner(null, this, this.Request.Form.Files[0], this._cache);
}

}
```

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

Web API with ASP.NET Core Web Application

Default ASP.NET Core Razor Pages template (Web Application) does not have MVC configuration to use Web API. So, need to add MVC options with App and Services using the below steps,

Open `Startup.cs` file.

Call `AddMvcOptions` with `services.AddRazorPages()` as in below code.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
    services.AddRazorPages().AddMvcOptions(option => option.EnableEndpointRouting =
false).AddNewtonsoftJson();
}
```

Call `UseMvc` with `app` in next of `app.UseRouting()` as in below code.

```
'csharp
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    ....
    ....
    app.UseRouting();
    app.UseMvc();
    ....
    ....
}
```

Set the service URL

To browse, open and save the reports in the application, set the WebAPI controller name to the `ServiceUrl` property of the web Report Designer. You can replace the following code in your web Report Designer page.

```
'html
<div style="height: 500px; width: 100%;">
<bold-report-designer id="designer" service-url="/api/ReportingAPI" ></bold-report-designer>
```

```
</div>
```

Run the Application

Run the sample application and you can see the web ReportDesigner on the page as displayed in the following screenshot.

![Report Designer demo](/static/assets/aspnet-core/report-designer/images/getting-started/Getting-Started_img9.png)

Dependencies

The ReportDesigner have the following list of internal and external dependencies to render the component.

Scripts

Internal dependencies

| Name | Details | CDN link |
|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| bold.reports.common.min.js | Common script for reporting widgets. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js |
| bold.reports.widgets.min.js | Supports Syncfusion widgets to render in HTML5 format and it contains the dependent Syncfusion controls that is common for both report designer | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js |

| | | |
|-------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | and viewer. | |
| bold.report-designer-widgets.min.js | Supports Syncfusion widgets to render in HTML5 format and it contains the dependent Syncfusion controls for report designer | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-widgets.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-widgets.min.js |
| ej.chart.min.js | Renders the chart gauge item. Add this script only if your report contains the chart gauge report item. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js |
| ej2-base.min.js | Renders the gauge item. Add this script only if your | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js |

| | | |
|-----------------------|------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | report contains the gauge report item. | |
| ej2-data.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js |
| ej2-pdf-export.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js |
| ej2-svg-base.min.js | Renders the gauge item. Add this script only if your report contains the gauge | Secured link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js Unsecured Link: https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js |

| | | |
|--------------------------|------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | report item. | |
| ej2-lineargauge.min.js | Renders the linear gauge item. Add this script only if your report contains the linear gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js |
| ej2-circulargauge.min.js | Renders the circular gauge item. Add this script only if your report contains the circular gauge report item. | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js |
| ej.map.min.js | Renders the map item. Add this script only if your report contains the map | Secured link: https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js Unsecured link: http://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js |

| | | |
|-----------------------------|------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | report item. | |
| bold.report-viewer.min.js | Used to preview the reports in report designer . | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js |
| bold.report-designer.min.js | Used to render the Syncfusion JavaScript Report Designer widget. | Secured Link: https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js Unsecured Link: http://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js |

External dependencies

| Name | Details | CDN link |
|---------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| jQuery 1.7.1 and later versions | Common jQuery script to render the Bold Reporting widgets | Secured Link: https://cdn.boldreports.com/external/jquery-1.10.2.min.js Unsecured Link: http://cdn.boldreports.com/external/jquery-1.10.2.min.js |
| jsrender | The script to render the templates in the browser. | Secured Link: https://cdn.boldreports.com/external/jsrender.min.js Unsecured Link: http://cdn.boldreports.com/external/jsrender.min.js |
| Codemirror | Report Designer requires the code mirror scripts to edit the SQL queries and Visual Basic code functions with syntax highlighter. codemirror.min.js | Secured Link: https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js Unsecured Link: http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js |

| | | |
|--|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | show-hint.min.js | http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js |
| | sql-hint.min.js | http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js |
| | sql.min.js | http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js |
| | vb.min.js | http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js |

Styles

Internal dependencies

| Name | Details |
|-----------------------------|-------------------------------------------------------------------------|
| bold.reports.all.min.css | It contains the styles and css references of dependent components. |
| bold.reportdesigner.min.css | It contains the styles and css references of Report Designer component. |

The CDN links for all supported themes are provided in the below table. Refer the following syntax:

[https://cdn.boldreports.com/\[version\]/content/\[theme-name\]/\[fileName\]](https://cdn.boldreports.com/[version]/content/[theme-name]/[fileName])

| Name | Details | CDN link |
|-----------------|----------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Material | Includes the CSS properties defined for the Syncfusion JavaScript Reporting component in material theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.css https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.css http://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css |
| Bootstrap-theme | Includes the CSS properties defined for the JavaScript Bold Reporting component in | Secured Link: https://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/bootstrap-theme/bold.reportdesigner.min.css |

| | | |
|------------------|------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | bootstrap theme. | |
| Office-365 | Includes the CSS properties defined for the JavaScript Bold Reporting component in office-365 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/office-365/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/office-365/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/office-365/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/office-365/bold.reportdesigner.min.css |
| High-contrast-01 | Includes the CSS properties defined for the JavaScript Bold Reporting component in high-contrast-01 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/high-contrast-01/bold.reportdesigner.min.css |
| High-contrast-02 | Includes the CSS properties defined for the JavaScript Bold Reporting component in high-contrast-02 theme. | Secured Link: https://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reports.all.min.css https://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reportdesigner.min.css Unsecured Link: http://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reports.all.min.css http://cdn.boldreports.com/2.2.32/content/high-contrast-02/bold.reportdesigner.min.css |

All the provided CDN links can be accessed either through [http](http://) or [https](https://).

External dependencies

| Name | Details | CDN link |
|------|---------|----------|
| | | |

| | | |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Codemirror | <p>Report Designer requires the code mirror styles to edit the SQL queries and Visual Basic code functions with syntax highlighter.</p> <p>codemirror.min.css show-hint.min.css</p> | <p>Secured Link:https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css</p> <p>Unsecured Link:http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css http://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css</p> |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Service side dependencies

Internal dependencies

Package | Purpose

Syncfusion.Compression.Net.Core | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core**, **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

Syncfusion.Pdf.Net.Core | Supports for exporting the report to a PDF.

Syncfusion.DocIO.Net.Core | Supports for exporting the report to a Word.

Syncfusion.XlsIO.Net.Core | Supports for exporting the report to an Excel.

Syncfusion.OfficeChart.Net.Core | It is a base library of the **Syncfusion.XlsIO.Net.Core package**.

External dependencies

Package | Purpose

Newtonsoft.Json | Serialize and deserialize the data or report for report designer. It is a mandatory package for the report designer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

System.Data.SqlClient | This is an optional package for the report viewer and designer. It should be referred in project when process the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0 .

Localization

Localization is the process of customizing the application for a given culture and locale - involving much more than the simple translation of text. In web report designer, localized strings appropriate to each culture are stored in separate files and loaded according to the UI culture setting.

By default report designer display strings in US English(en-US).

To render the static text with specific culture, refer the following corresponding culture script files and set culture name to the **locale** property of the Report Designer.


```
<li> ej.localetexts.fr-FR.min.js </li>
<li> ej.culture.fr-FR.min.js </li>
</ul>
```

Refer to the `ej.localetexts.fr-FR.min.js` script file from the `Scripts` folder of your application.

```
'html
<script src="https://cdn.boldreports.com/2.2.32/scripts/I10n/reportdesigner/ej.localetexts.fr-
FR.min.js"></script>
`
```

Refer to the `ej.culture.fr-FR.min.js` script file from the `Scripts` folder of your application.

```
'html
<script src="https://cdn.boldreports.com/2.2.32/scripts/i18n/ej.culture.fr-FR.min.js"></script>
`
```

To render the localization Report Designer, use the following code snippet.

The following code example illustrates how to set the `locale` property for Report Designer in the index page.

```
'html
<bold-report-designer id="designer" service-url="/api/ReportingAPI" locale="fr-FR"></bold-report-
designer>
`
```

The following code example illustrates how to add Report Designer localization scripts in the layout page.

```
'html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<title>@ ViewData["Title"] - Render Report Designer in French localization</title>
<environment include="Development">
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reports.all.min.css"
rel="stylesheet" />
```

```
<link href="https://cdn.boldreports.com/2.2.32/content/material/bold.reportdesigner.min.css"
rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.css"
rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.css"
rel="stylesheet" />

</environment>

</head>

<body style="height:100%;width:100%;padding:0;">

<environment include="Development">

<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>

<script src="https://cdn.boldreports.com/external/jsrender.min.js" type="text/javascript"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/codemirror.min.js"
type="text/javascript"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/show-hint.min.js"
type="text/javascript"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/addon/hint/sql-hint.min.js"
type="text/javascript"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.37.0/mode/sql/sql.min.js"
type="text/javascript"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.44.0/mode/vb/vb.min.js"
type="text/javascript"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js"></script>

<script
src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/common/bold.report-designer-
widgets.min.js"></script>

<!--Chart component script added before the Report Designer and viewer script to render report with
chart report item-->

<script src="https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js"
type="text/javascript"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/bold.report-designer.min.js"
type="text/javascript"></script>

<script src="https://cdn.boldreports.com/2.2.32/scripts/l10n/reportdesigner/ej.localetexts.fr-
FR.min.js"></script>
```

```
<script src="https://cdn.boldreports.com/2.2.32/scripts/i18n/ej.culture.fr-FR.min.js"></script>
</environment>
<div class="container body-content" style="height:100%;width:100%;">
@RenderBody()
<bold-script-manager></bold-script-manager>
</div>
@RenderSection("Scripts", required: false)
</body>
</html>
`
```

Integrate the component with Report Server

You can integrate Report Designer with Report Server to create, edit, browse and publish reports using the Report Server built-in API service.

The Report Designer requires the `serviceAuthorizationToken` to perform the API actions with Bold Report Server. Create a token for the user by using the server RESTful API, you can use the following code in `HomeController.cs` to generate the token.

```
`csharp
public partial class HomeController : Controller
{
    public IActionResult Index()
    {
        ViewBag.ServiceAuthorizationToken = this.GenerateToken("guest@boldreports.com", "demo");
        return View();
    }
    public IActionResult About()
    {
        ViewData["Message"] = "Your application description page.";
        return View();
    }
    public IActionResult Contact()
    {
        ViewData["Message"] = "Your contact page.";
        return View();
    }
}
```

```
}

public IActionResult Error()
{
    return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });
}

public string GenerateToken(string userName, string password)
{
    using (var client = new HttpClient())
    {
        client.DefaultRequestHeaders.Accept.Clear();
        var content = new FormUrlEncodedContent(new[]
        {
            new KeyValuePair<string, string>("grant_type", "password"),
            new KeyValuePair<string, string>("username", userName),
            new KeyValuePair<string, string>("password", password)
        });
        var result = client.PostAsync("https://on-premise-demo.boldreports.com/reporting/api/site/site1/token", content).Result;
        string resultContent = result.Content.ReadAsStringAsync().Result;
        var token = JsonConvert.DeserializeObject<Token>(resultContent);
        return token.token_type + " " + token.access_token;
    }
}

public class Token
{
    public string access_token { get; set; }
    public string token_type { get; set; }
    public string expires_in { get; set; }
    public string userName { get; set; }
    public string serverUrl { get; set; }
    public string password { get; set; }
}
```

Set the Bold Report Server built-in service URL to the `serviceUrl` property and assign the created token to `serviceAuthorizationToken` property. You can use the following code in your CSHTML page.

```
'html
<bold-report-designer id="designer" service-url="https://on-premise-
demo.boldreports.com/reporting/reportservice/api/Designer" service-authorization-
token="@ViewBag.ServiceAuthorizationToken" ajax-before-load="onAjaxRequest" ></bold-report-
designer>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'serverurl', Value: 'https://on-premise-
demo.boldreports.com/reporting/api/site/site1' });
}
</script>
'
```

Designing Reports

The Bold Report Designer provides an end-user documentation that describes how to interact with the Report Designer's UI and design an interactive reports. Refer the following user guide sections to get start with Bold Report Designer.

Connecting your data

A report must have a data source and dataset to include data. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

[Manage data source](#)

[Manage data set](#)

Transform your data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)

[Formatting columns](#)

[Query filter](#)

[Data set parameter](#)

[Query parameter](#)

[Configure expression columns](#)

Working with report parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it.

[Add report parameter](#)

[Edit report parameter](#)

[Delete report parameter](#)

[Cascading parameter](#)

[Multi-value parameter](#)

Embed images into the report

Supports to embed local or database images into the report and image data is stored within the report definition. The embedded images in a report are listed in the **Image Manager panel**.

[Embed images into the report](#)

Interacting with report design surface

WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

[Interacting with report design surface](#)

Report items

Offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

[Configure report items](#)

Customize appearance

The Properties panel shows all the properties of the selected section, report items or the report itself to customize the appearance of the report.

[Customize appearance](#)

Report design settings

[Configure design settings](#)

Shape report data

[Filter data](#)

[Group data](#)

[Sort data](#)

[Format data](#)

Interactive features

[Hyperlink](#)

[Drilldown](#)

[Interactive sorting](#)

Working with Expressions

Expressions are used to specify criteria for retrieving and formatting data, creating calculated fields and calculating summaries, conditionally shaping data and changing a report control's appearance.

[Expressions builder](#)

[Open report](#)

[Open report](#)

[Save report](#)

[Save report](#)

[Preview report](#)

[Preview report](#)

Migrate Report Designer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Designer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Designer from Syncfusion Essential Studio release version to Bold Reports version of ASP.NET Core Report Designer application.

Scripts

| Old Scripts | New Scripts | Description |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej.web.all.min.js | bold.reports.common.min.js bold.reports.widgets.min.js bold.report-designer-widgets.min.js ej.chart.min.js ej2-base.min.js ej2-data.min.js ej2-pdf-export.min.js ej2-svg-base.min.js ej2-lineargauge.min.js ej2-circulargauge.min.js ej.map.min.js bold.report-viewer.min.js | <p>We have removed the dependency scripts for report designer from existing ej.web.all.min.js script and provided the dependency scripts as below:</p> <p>bold.reports.common.min.js - Common script for reporting widgets.</p> <p>bold.reports.widgets.min.js - It contains the scripts of dependent controls that are common for both Report Designer and Report Viewer.</p> <p>bold.report-designer-widgets.min.js - It contains the scripts of Report Designer dependent controls.</p> <p>ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart</p> |

| | | |
|---------------------------------------|------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | <p>report item.</p> <p><code>ej2-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-data.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-pdf-export.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-svg-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-lineargauge.min.js</code> - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p><code>ej2-circulargauge.min.js</code> - Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Previews the reports designed with Report Designer.</p> |
| <code>ej.reportdesigner.min.js</code> | <code>bold.report-designer.min.js</code> | Renamed the report designer script file and it used to render the Bold Report Designer widget. |

Styles

| Old Scripts | New Scripts | Description |
|--------------------|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej.web.all.min.css | bold.reports.all.min.css | We have removed the dependent controls styles for report designer from existing ej.web.all.min.css script and provided the dependent controls styles as bold.reports.all.min.css. |

Remove the old scripts and styles references from existing application, then add the new scripts and styles references based on above script name changes.

Assemblies

| Purpose | Old Assembly Name | New Assembly Name | Description |
|-----------------------|------------------------------------------------------------------------------------------------------|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Base Assemblies | Syncfusion.EJ.ReportDesigner.dll Syncfusion.EJ.ReportViewer.dll Syncfusion.Report.Portable.dll | BoldReports.Web.dll | The Syncfusion.EJ.ReportDesigner.dll, Syncfusion.EJ.ReportViewer.dll, and Syncfusion.Report.Portable.dll assemblies have been combined as BoldReports.Web.dll. It builds the server-side implementations for both Report Designer and Report Viewer components. |
| Tag Helper Assemblies | Syncfusion.EJ.dll Syncfusion.EJ.AspNetCore.dll | BoldReports.AspNetCore.dll | The Syncfusion.EJ.dll, Syncfusion.EJ.AspNetCore.dll assemblies have been combined as BoldReports.AspNetCore.dll. |

Packages

| Purpose | Old Packages | New Packages |
|--------------------|----------------------------------------------------------------------------------------------------------------|------------------------|
| Server Side Helper | Syncfusion.EJ.ReportDesigner.AspNetCore Syncfusion.EJ.ReportViewer.AspNetCore Syncfusion.Report.Net.Core | BoldReports.Net.Core |
| Tag Helper | Syncfusion.EJ.AspNetCore | BoldReports.AspNetCore |

Based on above assembly and namespace changes, modify the Report Designer **Web API Controller** in your application.

Tag helper

Modify the tag helpers in the `_ViewImports.cshtml` page.

| Old code snippet | New code snippet |
|------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| <code>@using Syncfusion.JavaScript @addTagHelper *, Syncfusion.EJ</code> | <code>@using BoldReports.TagHelpers @addTagHelper *, BoldReports.AspNetCore</code> |

Configure script manager

The ASP.NET core reporting components tag prefix has been changed from `ej` to `sf`.

| Old code snippet | New code snippet |
|-----------------------------------------------------------------------|---------------------------------------------------------------------------|
| <code><ej-script-manager></ej-script- manager></code> | <code><bold-script-manager></bold-script- manager></code> |

Control initialization

| Old code snippet | New code snippet |
|--------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| <code><ej-report-designer id="designer"></ej-report-designer></code> | <code><bold-report-designer id="designer"></bold-report-designer></code> |

NuGet Packages for ASP.NET Core

Refer to the following steps to configure Bold Reporting tools NuGet packages for ASP.NET Core application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting tools NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: [NuGet.org](#)

Source: <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/aspnet-core/report-designer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reporting tools NuGet packages are shipped into our Report Platform SDK build. To configure the packages from Bold Reports installed location, use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages.

![Offline NuGet Configure](/static/assets/aspnet-core/report-designer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

Installing NuGet packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**.

Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages (**BoldReports.Net.Core** or **BoldReports.AspNet.Core**), and then click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

install specified package in default project

Upgrading NuGet packages

install specified package in default project

Install-Package <Package Name>

install specified package in default project with specified package source

Install-Package <Package Name> -Source <Source Location>

install specified package in specified project

Install-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

install specified package in default project

Install-Package BoldReports.Net.Core

install specified package in default project with specified Package Source

Install-Package BoldReports.Net.Core –Source “<https://api.nuget.org/v3/index.json>”

install specified package in specified project

Install-Package BoldReports.Net.Core -ProjectName BoldReportingApplication

Upgrading NuGet packages

[Upgrading using NuGet Package Manager](#)

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution...**

Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

[Upgrading using Package Manger Console](#)

To update the installed Bold Reporting tools NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

`cmd

Update specific NuGet package in default project

Update-Package <Package Name>

Update all the packages in default project

Update-Package

Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

Update specified Bold Reporting NuGet package

Update-Package BoldReports.Net.Core

Update specified package in default project with specified Package Source

Update-Package BoldReports.Net.Core –Source “<https://api.nuget.org/v3/index.json>”

Update specified package in specified project

Update-Package BoldReports.Net.Core -ProjectName BoldReportingApplication

Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing nuget.exe to latest version use the following command:

`cmd

nuget update -self

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting tools NuGet packages.

update all NuGet packages from config file

Normal layout

`cmd

[update all NuGet packages from config file](#)

nuget update <configPath> [options]

[update all NuGet packages from specified Packages Source](#)

nuget update -Source <Source Location> [optional]

configPath is optional. It identifies the package.config or solutions file lists the packages utilized in the project.

For example:

`cmd

[Update all NuGet packages from config file](#)

nuget update "C:\Users\SyncfusionApplication\package.config"

[Update all NuGet packages from specified Packages Source](#)

nuget update -Source "https://api.nuget.org/v3/index.json"

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

Responsive layout rendering of ASP.NET Core Report Designer

Report Designer will adaptively render itself with optimal user interfaces for tablet or desktop form factors. It has built-in responsive support, so that it will adjust automatically based on the browser viewport. Setting width to 100% is simply enough to make it responsive. This helps your application to scale elegantly on all form factors with ease.

Normal layout

The following output shows the normal layout rendering of Report Designer tool bar items.

![Rendering of Report Designer tool bar items in normal layout](/static/assets/aspnet-core/report-designer/images/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Designer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/aspnet-core/report-designer/images/responsive-layout.png)

Samples and Demos

Browse and explore our ready-to-use the designer samples, online and offline demos.

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view the ASP.NET Core Report Designer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view the GitHub Report Designer demo samples.

Supported Browsers

This document provides the list of web browsers supported by the Web Report Designer

| Desktop Browsers | Version |
|-------------------|---------|
| Internet Explorer | 11.0+ |
| Microsoft Edge | Current |
| Firefox | 22+ |
| Google Chrome | 17+ |
| Opera | 12+ |
| Safari | 5+ |

How to queries for Bold Reports Report Designer

This section helps to get the answer for the frequently asked how to queries in Bold Reports Report Designer.

[Open server report using report path on opening Report Designer](#)

[How to add datasource and dataset for Report Designer from application?](#)

How to open server report using report path at the time of opening the Report Designer

Find the following steps to open server report using report path on opening Report Designer.

Create a function and bind it with the `create` API in `Index.cshtml` file as like in below code snippet.

```
'html
<bold-report-designer id="designer" create="controlInitialized"></bold-report-designer>
<script type="text/javascript">
function controlInitialized(args) {
...
}
```

```
</script>
`
```

Use the `openReport` method in the function along with report path that was previously created, as like in below code snippet.

```
`javascript
function controlInitialized(args) {
    var designer = $('#designer').data('boldReportDesigner');
    designer.openReport("/Sample Reports/Sales Report");
}
`
```

How to add the data source and dataset for Report Designer from application

You can add the data for reports in the Report Designer from application level on initializing the Report Designer. This can be achieved using the `addDataSource` and `addDataSet` functions, by which you can add the data source and dataset for the reports at the time of initialization of Report Designer.

Find the following steps to add the data source and dataset for Report Designer from application.

Create a function and bind it with the `create` API in `Index.cshtml` file as shown in the following code snippet.

```
`html
<bold-report-designer id="designer" create="controlInitialized"></bold-report-designer>
<script type="text/javascript">
function controlInitialized(args) {
    ...
}
</script>
`
```

Use the `addDatasource` method to add the data source in Report Designer as shown in the following code snippet,

```
`javascript
var datasource =
{
    type:'BoldReports.RDL.DOM.DataSource',
```

```

Name:'DataSource1',
Transaction:false,
DataSourceReference:null,
SecurityType:'DataBase',
ConnectionProperties:{
    type:'BoldReports.RDL.DOM.ConnectionProperties',
    ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
    EmbedCredentials:false,
    DataProvider:'SQL',
    IsDesignState:false,
    IntegratedSecurity:false,
    UserName:'',
    PassWord:'',
    Prompt:'Specify the Username and password for DataSource DataSource1',
    CustomProperties:[]
}
};

function controlInitialized(args) {
    var designer = $('#designer').data('boldReportDesigner');
    designerObj.addDataSource(datasource);
}

`
```

Use the `addDataSet` method to add dataset in Report Designer as shown in the following code snippet,

```

`javascript
var dataset =
{
    type:'BoldReports.RDL.DOM.DataSet',
    Name:'DataSet1',
    Fields:[
        { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},

```

```
{ type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},  
{ type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },  
{ type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null }  
],  
Query: {  
    type: "BoldReports.RDL.DOM.Query",  
    CommandText: "SELECT  
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[HumanResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM  
[HumanResources].[Department]",  
    CommandType: 0,  
    DataSourceName: "DataSource1",  
    QueryDesignerState: {  
        type: "BoldReports.RDL.DOM.QueryDesignerState",  
        Expressions: null,  
        Filters: null,  
        Joins: null,  
        StoredProcedure: null,  
        Tables: [  
            {  
                type: "BoldReports.RDL.DOM.Table",  
                Columns: [  
                    { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "DepartmentID" },  
                    { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "Name" },  
                    { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false, IsSelected: true, Name: "GroupName" },  
                    { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
```

```
    IsSelected: true, Name: "ModifiedDate"
  }
],
Name: "Department",
Schema: "HumanResources",
SchemaLevels: [
  { Name: "HumanResources", SchemaType: "Schema" },
  { Name: "Tables", SchemaType: "Category" },
  { Name: "Department", SchemaType: "Table" }
]
}
]
},
QueryParameters: [],
Timeout: 0
},
CaseSensitivity:0,
Collation:null,
AccentSensitivity:0,
KanatypeSensitivity:0,
WidthSensitivity:0,
Filters:[],
SharedDataSet:null,
InterpretSubtotalsAsDetails:0,
DataSetInfo:null,
DataSetObject:null
};
function controlInitialized(args) {
var designer = $('#designer').data('boldReportDesigner');
designerObj.addDataSource(dataset);
}
`
```

Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

Breaking Changes

When you upgrade from previous versions of Bold Reports to 2.2.23, there are few breaking changes. The following section explains the breaking changes for Bold Reports version 2.2.23.

Designer resource read and write API

The resource write and read API's in `IReportDesignerController` are modified to simplify the resource write and read actions with Bold Report Designer.

Below are the new API details tabulated against old API's.

| Old API | New API | Description |
|-------------|---------|-----------------------------------------------------|
| UploadFile | SetData | Writes the designer resource into storage location. |
| GetFilePath | GetData | Reads the designer resource from storage location. |
| GetFiles | | |
| GetFile | | |

Parameters

`SetData`

| Parameter Name | Type | Description |
|----------------|---------------|------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |
| itemData | object | Gets the resource data. |
| | Property Name | |
| | Data | |

The `itemData` parameter is an object containing two properties:

- `Data`: Type: byte array. Description: Gets the resource data as byte array. To write an external resource into storage location, pass the resource data in this property.
- `PostedFile`: Type: IFormFile. Description: Gets the uploaded resource data as

| | | | | |
|--------------|--------|--|-------------------------------------------------------------------------------------------------|--|
| | | | IFormFile. The resource data uploaded within report designer will be received in this property. | |
| errorMessage | string | | Returns the error message, if the write action is failed. | |

GetData

| Parameter Name | Types | Description |
|----------------|--------|------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |

Return Type

| API Name | Return Type | |
|---------------|-------------|----------------------------------------------------------|
| SetData | boolean | |
| GetData | object | |
| Property Name | Type | Description |
| Data | byte array | Contains the requested resource data as byte array. |
| errorMessage | string | Returns the error message, if the read action is failed. |

Code snippet

| Old snippet | New snippet |
|---------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| UploadFilecsharp public bool UploadFile(System.Web.HttpPostedFile httpPostedFile){ //Implement resource write actions} | SetDatacsharp public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage){ //Implement resource write actions} |
| GetFilescsharp public List GetFiles(FileType fileType){ //Implement resource read actions} | |

| | |
|---------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <code>GetFilePath</code> csharppublic string GetFilePath(string fileName){ //Implement actions to get file path} | <code>GetData</code> csharppublic ResourceInfo GetData(string key, string itemId){ //Implement resource read actions} |
| <code>GetFile</code> csharppublic FileModel GetFile(string filename, bool isOverride){ //Implement resource read actions} | |

Reporting tools for ASP.NET MVC

Enterprise-class reporting tools for ASP.NET MVC development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your MVC applications.

How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Reporting tools for ASP.NET MVC

Enterprise-class reporting tools for ASP.NET MVC development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your MVC applications.

How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

System Requirements

This topic describes the software and hardware requirements for setting up development environment of Bold Reports ASP.NET MVC.

Supported Operating Systems

Windows 7+, 8+

Windows Server 2008 R2+

Hardware Requirements

The following hardware requirements are necessary setting up development environment of Bold Reports ASP.NET MVC:

1 GHZ or faster, 32 bit or 64 bit processor.

1 GB RAM for 32 bit or 2 GB RAM for 64 bit.

Software Requirements

The following software requirements are necessary setting up development environment of Bold Reports ASP.NET MVC:

Microsoft Visual Studio 2010 or later

Internet Information Services (IIS) 7.0+

Framework

The below tool is required for Bold Reports ASP.NET MVC.

.NET Framework 4.5 or higher

Browser Compatibility

IE 9+

Microsoft Edge

Mozilla Firefox 22+

Chrome 17+

Opera 12+

Safari 5+

See Also

[Licensing procedure for deployment](#)

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Microsoft Excel, CSV, Microsoft Word, and HTML formats. Some of the key features are,

Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.

Easily customize each element of Report Viewer and provide events for report processing customization.

Display ssrs rdl report in Bold Reports ASP.NET MVC Report Viewer

This section explains you the steps required to create your first ASP.NET MVC reporting application to display already created SSRS RDL report in Bold Reports ASP.NET MVC Report Viewer without using a Report Server.

Create ASP.NET MVC 5 application

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![Asp.Net MVC5 project template](/static/assets/aspnet-mvc/report-viewer/images/getting-started/aspnetmvc5-template.png)

Choose **MVC** and **Web API**, and then click **OK**.

![Asp.Net MVC5 web application template](/static/assets/aspnet-mvc/report-viewer/images/getting-started/aspnetmvc5-config-template.png)

Configure Report Viewer in an application

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution**.

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web**, **BoldReports.JavaScript** and **BoldReports.Mvc5** NuGet packages, and install them in your MVC application.

Package | Purpose

PostReportAction | Action (HttpPost) method for posting the request in report process.

OnInitReportOptions | Report initialization method that occurs when the report is about to be processed.

OnReportLoaded | Report loaded method that occurs when the report and sub report start loading.

GetResource | Action (HttpGet) method to get resource for the report.

ReportHelper

The class **ReportHelper** contains helper methods that help to process a Post or Get request from the Report Viewer control and return the response to the Report Viewer control. It has the following methods:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

Add Web API Controller

Right-click **Controller** folder in your project and select **Add > New Item** from the context menu.

Select **Web API Controller Class** from the listed templates and name it as **ReportViewerController.cs**.

![Provide controller name](/static/assets/aspnet-mvc/report-viewer/images/getting-started/aspnet-mvc-webapi-template.png)

Click **Add**.

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
public class ReportViewerController : ApiController, IReportController
```

```
{  
// Post action for processing the RDL/RDLC report  
public object PostReportAction(Dictionary<string, object> jsonResult)  
{  
    return ReportHelper.ProcessReport(jsonResult, this);  
}  
  
// Get action for getting resources from the report  
[System.Web.Http.ActionName("GetResource")]  
[AcceptVerbs("GET")]  
public object GetResource(string key, string resourcetype, bool isPrint)  
{  
    return ReportHelper.GetResource(key, resourcetype, isPrint);  
}  
  
// Method that will be called when initialize the report options before start processing the report  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    // You can update report options here  
}  
  
// Method that will be called when reported is loaded  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    //You can update report options here  
}  
}  
`
```

Add routing information

To configure routing to include an action name in the URI, open the `WebApiConfig.cs` file and change the `routeTemplate` in the `Register` method as follows.

```
'csharp  
public static class WebApiConfig  
{  
    public static void Register(HttpConfiguration config)  
    {  
        // Web API configuration and services
```

Load SSRS Report Server reports

Set report path and service URL

```
// Web API routes
config.MapHttpAttributeRoutes();
config.Routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{action}/{id}",
    defaults: new { id = RouteParameter.Optional }
);
}
}
`
```

Set report path and service URL

To render the reports available in the application, set the `ReportPath` and `ReportServiceUrl` properties of the Report Viewer. You can replace the following code in your Report Viewer page.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
)
`
```

The report path property is set to the RDL report that is added to the project `Resources` folder.

Preview the report

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Sales Order Detail Report](/static/assets/aspnet-mvc/report-viewer/images/getting-started/sales-order-detail.png)

See Also

[Render report with data visualization report items](#)

[Create RDLC report](#)

[List of SSRS server versions are supported in Bold Reports](#)

Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports, set the `ReportServerUrl`, `ReportPath`, and `ReportServiceUrl` properties as shown in the following code snippet.

```
`js
@(Html.Bold().ReportViewer("viewer")`
```

```
.ReportServiceUrl("/api/SSRSReports")
.ReportServerUrl("http://<servername>/reportserver$instanceName")
.ReportPath("/SSRSSamples2/Territory Sales new")
)
`
```

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of `/folder name/report name`.

Network credentials for SSRS

The network credentials are required to load specified SSRS report from the specified SSRS Report Server using the Report Viewer. Specify the `ReportServerCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server credential
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");
}
```

Set data source credential to shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS Server. If the report has any data source that uses credentials to connect with the database, then you should specify the `DataSourceCredentials` for each report data source to establish database connection.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Data source credentials should be added to the shared data sources that do not have credentials in the connection strings.

Render linked reports

You can render a linked report that points to an existing report, which is published in the SSRS Report Server. You can set the parameter, data source, credential, and other properties like normal SSRS reports using the Report Viewer.

```
'js
```

```
@(Html.Bold()).ReportViewer("viewer")
    .ReportServiceUrl("/api/SSRSReports")
    .ReportServerUrl("http://<servername>/reportserver$instanceName")
    .ReportPath("/SSRSSamples/Territory Sales_Link")
)
```

The Territory Sales_Link is a linked report created for the Territory Sales report that is already available in the SSRS Report Server.

Load SharePoint Server reports

To render SharePoint server reports, set the ReportServerUrl, ReportPath, and ReportServiceUrl properties as shown in the following code snippet.

```
'js
```

```
@(Html.Bold()).ReportViewer("viewer")
    .ReportServiceUrl("/api/SharePointReports")
    .ReportServerUrl("http://<servername>/reportserver$instanceName")
    .ReportPath("http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl")
)
```

In SharePoint integrated mode, the ReportServerUrl will be same as your site URL. The ReportPath is relative to the Report Server URL with the file extension.

Forms credential for SharePoint Server

The forms credentials are required to load the SharePoint integrated SSRS report from the specified SharePoint integrated SSRS Report Server using the Report Viewer. Specify the ReportServerFormsCredential property in the Web API Controller OnInitReportOptions method.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential for server
    reportOption.ReportModel.ReportServerFormsCredential = new ReportServerFormsCredential("ssrs",
        "RDLReport1");
```

Load Bold Report Server reports

Set data source credential to shared data sources

}

Set data source credential to shared data sources

The shared data source credentials can be added to the `DataSourceCredentials` property to connect with the database.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential and data source credentials
    reportOption.ReportModel.ReportServerFormsCredential = new ReportServerFormsCredential("ssrs",
    "RDLReport1");
    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Data source credentials should be added to shared data sources that do not have credentials in the connection strings.

Load Bold Report Server reports

You can render the Bold Report Server reports in the Report Viewer easily without creating a Web API service. Bold Report Server provides the built-in Web API service that helps you to display the server reports.

The Report Viewer requires the `ServiceAuthorizationToken` to connect and download the items from the Bold Report Server.

To load the Report Server reports, follow these steps:

Create a token for users by using the server RESTful API. The following code is used to generate the token.

```
'csharp
public partial class HomeController : Controller
{
    public ActionResult Index()
    {
        return View();
    }
    public string GenerateToken( string userName, string password)
    {
```

```

using (var client = new HttpClient())
{
    client.DefaultRequestHeaders.Accept.Clear();
    var content = new FormUrlEncodedContent(new[]
    {
        new KeyValuePair<string, string>("grant_type", "password"),
        new KeyValuePair<string, string>("username", userName),
        new KeyValuePair<string, string>("password", password)
    });
    var result = client.PostAsync("https://on-premise-
        demo.boldreports.com/reporting/api/site/site1/token", content).Result;
    string resultContent = result.Content.ReadAsStringAsync().Result;
    var token = JsonConvert.DeserializeObject<Token>(resultContent);
    return token.token_type + " " + token.access_token;
}
}
}

public class Token
{
    public string access_token { get; set; }
    public string token_type { get; set; }
    public string expires_in { get; set; }
    public string userName { get; set; }
    public string serverUrl { get; set; }
    public string password { get; set; }
}
`
```

Set the Bold Report Server built-in service URL to the `ReportServiceUrl` property.

Assign the created token to `ServiceAuthorizationToken` and `ReportPath` of a report deployed on the server. You can use the following complete code in your `index.cshtml` page.

```

`js
@{
```

```

var controller = ViewContext.Controller as ReportViewer_MVC5.Controllers.HomeController;
var apiToken = controller.GenerateToken("https://on-premise-
demo.boldreports.com/reporting/api/site/site1", "guest@boldreports.com", "demo");
}

@(Html.Bold().ReportViewer("viewer")
    .ServiceAuthorizationToken(@apiToken)
    .ReportServiceUrl(@"https://on-premise-demo.boldreports.com/reporting/reportservice/api/Viewer")
    .ReportPath("/Sample Reports/Sales Order Detail")
    .Parameters(param => { param.Name("SalesOrderNumber").Labels(new List<string>() { "SO50751" })
        }.Values(new List<string>() { "SO50751" }).Add(); })
    .AjaxBeforeLoad("onAjaxRequest")
)

<script type="text/javascript">
function onAjaxRequest(args) {
    args.headers.push({ Key: 'serverurl', Value: 'https://on-premise-
    demo.boldreports.com/reporting/api/site/site1' });
}
</script>
'

```

You can also load the report using GUID instead of report location. Set the GUID of the report in the **ReportPath** property as `ReportPath("91f24bf1-e537-4488-b19f-b37f77481d00")`.

Render RDLC report

The data binding support, allows you to view RDLC reports that exist on the local file system with JSON array and custom business object data collection. The following steps demonstrates how to render a RDLC report with JSON array and custom business object data collection.

Add the RDLC report `product-list.rdlc` from Bold Reports installation location to your application **Resources** folder. For more information, see [Samples and demos](#).

Bind data source at client side

Set the RDLC report path to the **ReportPath** property.

Assign the **ProcessingMode** property to `ProcessingMode.Local`.

Bind the data from `viewdata` collection to the **DataSources** property as shown in following code.

```

`js
@(Html.Bold().ReportViewer("viewer"))

```

```
.ProcessingMode(BoldReports.ReportViewerEnums.ProcessingMode.Local)
.ReportPath("~/Resources/product-list.rdlc")
.ReportServiceUrl("/api/ReportViewer")
.DataSources(ds => ds.Name("list").Value(ViewData["DataSource"]).Add())
)
```

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
public partial class HomeController : Controller
{
    public ActionResult Index()
    {
        ProductList productList = new ProductList();
        ViewData["DataSource"] = productList.GetData();
        return View();
    }
}

public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public IList GetData()
    {
        List<ProductList> dataList = new List<ProductList>();
        ProductList data = null;
        data = new ProductList() {ProductName = "Baked Chicken and Cheese",OrderId = "323B60",Price = 55,Category = "Non-Veg",Ingredients = "grilled chicken, corn and olives.",ProductImage = "" };
        dataList.Add(data);
    }
}
```

```

data = new ProductList() {ProductName = "Chicken Delite",OrderId = "323B61",Price = 100,Category =
"Non-Veg",Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",ProductImage = ""};

dataList.Add(data);

data = new ProductList() {ProductName = "Chicken Tikka",OrderId = "323B62",Price = 64,Category =
"Non-Veg",Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",ProductImage = ""};

dataList.Add(data);

return dataList;

}
}
`
```

[Bind data source in Web API controller](#)

The following steps help you to configure the Web API to render the RDLC report with business object data collection.

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```

`csharp
public class ProductList
{
    public string ProductName { get; set; }

    public string OrderId { get; set; }

    public double Price { get; set; }

    public string Category { get; set; }

    public string Ingredients { get; set; }

    public string ProductImage { get; set; }

    public static IList GetData()
    {
        List<ProductList> dataList = new List<ProductList>();

        ProductList data = null;

        data = new ProductList() {ProductName = "Baked Chicken and Cheese",OrderId = "323B60",Price =
55,Category = "Non-Veg",Ingredients = "grilled chicken, corn and olives.",ProductImage = "" };

        dataList.Add(data);

        data = new ProductList() {ProductName = "Chicken Delite",OrderId = "323B61",Price = 100,Category =
"Non-Veg",Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",ProductImage = ""};

        dataList.Add(data);
    }
}
```

```

data = new ProductList() {ProductName = "Chicken Tikka",OrderId = "323B62",Price = 64,Category =
"Non-Veg",Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",ProductImage = ""};

dataList.Add(data);

return dataList;

}
}

'

```

Set the `ProcessingMode` to `ProcessingMode.Local` and `ReportPath` to the RDLC report location.

Bind the business object data values collection by adding new item to the `DataSources` as in the following code snippet.

```

`csharp

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;
    reportOption.ReportModel.ReportPath =
        System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/product-list.rdlc");
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
'
```

Here the `Name` is case sensitive and it should be same as in the data source name in the report definition.

The `Value` accepts `IList`, `DataSet`, and `DataTable` inputs.

[Load report as stream](#)

To load report as a stream, create a report stream using the `FileStream` class and assign the report stream to the `Stream` property.

```

`csharp

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    // Opens the report from application Resources folder using FileStream
    FileStream reportStream = new
        FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/product-list.rdlc"),
        FileMode.Open, FileAccess.Read);
    reportOption.ReportModel.Stream = reportStream;
}
```

Render subreport

Change subreport path

```
reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;  
reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",  
Value = ProductList.GetData() });  
}  
,
```

In the above code, `product-list.rdlc` report is loaded from the `Resources` folder location.

Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

Add the sub report and main reports to the application `Resources` folder. In this tutorial, the already created reports are used. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section.

Download the `side-by-side-main-report.rdl`, `side-by-side-sub-report.rdl` reports from [here](#). You can add a report from the Syncfusion installation location. For more information, refer to [Samples and demos](#).

Set the `ReportPath` and `ReportServiceUrl` properties of the Report Viewer as in the following code snippet.

The following code example demonstrates how to load a subreport in the Report Viewer at client side.

```
`js  
@(Html.Bold().ReportViewer("viewer")  
.ReportPath("~/Resources/side-by-side-main-report.rdl")  
.ReportServiceUrl("/api/ReportViewer")  
)  
,
```

Change subreport path

To change the subreport file path, set the `Stream` property of `SubReportModel` in the `OnInitReportOptions` method.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    string reportPath = System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/sub-report-  
detail.rdl");  
    if (reportOption.SubReportModel != null)  
    {
```

Render subreport

Set subreport parameter

```
FileStream SubStream = new FileStream(reportPath, FileMode.Open, FileAccess.Read);
reportOption.SubReportModel.Stream = SubStream;
}
}
`
```

Set subreport parameter

You can change the parameter default values of a subreport in the `OnReportLoaded` method of the Web API Controller as given in the following code snippet.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (reportOption.SubReportModel != null)
{
reportOption.SubReportModel.Parameters = new BoldReports.Web.ReportParameterInfoCollection();
reportOption.SubReportModel.Parameters.Add(new BoldReports.Web.ReportParameterInfo()
{
Name = "SalesPersonID",
Values = new List<string>() { "2" }
});
}
}
```

Modify subreport data source connection string

You can change the credential and connection information of the data sources used in the subreport using the `SubReportModel` in the `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
if (reportOption.SubReportModel != null)
{
reportOption.SubReportModel.DataSourceCredentials = new
List<BoldReports.Web.DataSourceCredentials>();
reportOption.SubReportModel.DataSourceCredentials.Add(new
BoldReports.Web.DataSourceCredentials("NorthWind", "Data
```

Render subreport

Set subreport data source

```
Source=dataplatformdemodata.syncfusion.com;Initial Catalog=Northwind;user id=demoreadonly@data-platform-demo;password=N@c=Y8s*1&dh");
```

```
}
```

```
}
```

```
'
```

Set subreport data source

You can bind local business object data source collection only for RDLC reports. To specify data source of a RDLC subreport, set the `ReportDataSource` property in the `OnReportLoaded` method.

The RDL report has the connection information in report definition itself, so no need to bind data source.

Add the RDLC sub report and main reports to your application `Resources` folder. For more information, see [Samples and demos](#).

Set the `ReportPath`, `ProcessingMode`, and `ReportServiceUrl` properties of the Report Viewer as shown in following code snippet.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ProcessingMode(BoldReports.ReportViewerEnums.ProcessingMode.Local)
    .ReportServiceUrl("/api/SubreportDataSources")
    .ReportPath("~/Resources/product-list.rdlc")
)
```

Create a class and methods that returns business object data collection. Use the following code in the application Web API Service.

```
`csharp
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }
    public static IList GetData()
```

Render subreport

Set subreport data source

```
{  
List<ProductList> datas = new List<ProductList>();  
ProductList data = null;  
data = new ProductList()  
{  
ProductName = "Baked Chicken and Cheese",  
OrderId = "323B60",  
Price = 55,  
Category = "Non-Veg",  
Ingredients = "grilled chicken, corn and olives.",  
ProductImage = ""  
};  
datas.Add(data);  
data = new ProductList()  
{  
ProductName = "Chicken Delite",  
OrderId = "323B61",  
Price = 100,  
Category = "Non-Veg",  
Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",  
ProductImage = ""  
};  
datas.Add(data);  
data = new ProductList()  
{  
ProductName = "Chicken Tikka",  
OrderId = "323B62",  
Price = 64,  
Category = "Non-Veg",  
Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",  
ProductImage = ""  
};  
datas.Add(data);
```

Render subreport

Load subreport stream

```
return datas;  
}  
}  
`
```

Bind the business object data values collection to the subreport by adding a new item to the **DataSources** as shown in the following code snippet.

```
`csharp  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    //Assigning the data source for 'Product List.rdlc'  
    if (reportOption.SubReportModel != null)  
    {  
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();  
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =  
            "list", Value = ProductList.GetData() });  
    }  
}
```

The data source name is case sensitive, it should be same as in the report definition.

[Load subreport stream](#)

To load subreport as stream, set the **Stream** property in the **OnInitReportOptions** method.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    if (reportOption.SubReportModel != null)  
    {  
        FileStream reportStream = new  
        FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/product-list.rdlc"),  
        FileMode.Open, FileAccess.Read);  
        reportOption.SubReportModel.Stream = reportStream;  
    }  
}  
  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{
```

| | |
|-------------------|-------------------------|
| Report parameters | Set parameter at client |
|-------------------|-------------------------|

```
if (reportOption.SubReportModel != null)
{
    reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
    reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
"list", Value = ProductList.GetData() });
}
}
`
```

Report parameters

Provides property options to pass or set report parameters default values at run time using the **parameters** property. You can set the report parameters while creating the Report Viewer control in a script or in the Web API Controller.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#).

Set parameter at client

To set parameter default value in the client side, The following code example illustrates how to set report parameter,

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ReportServiceUrl("/api/ReportViewer")
    .Parameters(param => { param.Name("SalesOrderNumber").Labels(new List<string>() { "SO50755" })
        .Values(new List<string>() { "SO50755" }).Add(); })
)
`
```

Set parameters in Web API Controller

To set parameter default value in Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    List<BoldReports.Web.ReportParameter> userParameters = new
    List<BoldReports.Web.ReportParameter>();
    userParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = "SalesOrderNumber",
```

```
Values = new List<string>() { "SO50755" }  
});  
reportOption.ReportModel.Parameters = userParameters;  
}  
`
```

Get report parameter

The `ReportHelper` class provides methods to get the report parameters used in the report. The following helper methods used to get parameter with or without values.

Methods | Description

`GetParameters` | Returns the parameters used in the current report without the processed values.

`GetParametersWithValues` | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

`csharp

```
public class ReportViewerController : ApiController, IReportController  
{  
    Dictionary<string, object> jsonArray = null;  
  
    public object PostReportAction(Dictionary<string, object> jsonResult)  
    {  
        jsonArray = jsonResult;  
        return ReportHelper.ProcessReport(jsonResult, this);  
    }  
  
    [System.Web.Http.ActionName("GetResource")]  
    [AcceptVerbs("GET")]  
  
    public object GetResource(string key, string resourcetype, bool isPrint)  
    {  
        return ReportHelper.GetResource(key, resourcetype, isPrint);  
    }  
  
    public void OnInitReportOptions(ReportViewerOptions reportOption)  
    {  
        // You can update report options here  
    }  
  
    public void OnReportLoaded(ReportViewerOptions reportOption)  
    {
```

Report interaction events

Report loaded

```
var reportParameters = ReportHelper.GetParameters(jsonArray, this);
List<BoldReports.Web.ReportParameter> setParameters = new
List<BoldReports.Web.ReportParameter>();
if (reportParameters != null)
{
foreach (var rptParameter in reportParameters)
{
setParameters.Add(new BoldReports.Web.ReportParameter()
{
Name = rptParameter.Name,
Values = new List<string>() { "SO50755" }
});
}
reportOption.ReportModel.Parameters = setParameters;
}
```

Report interaction events

You can handle the report interaction events with reports using the following events.

ReportLoaded

ReportError

ShowError

Drill through

Hyperlink

Report loaded

The **ReportLoaded** event occurs after the report is loaded and it is ready to start the processing. You can handle the event and specify the data source and parameters at client side. The following sample code loads a report by assigning the report data source input in the **ReportLoaded** event.

In this tutorial, the **product-list.rdlc** report is used. You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

```

`js
@(Html.Bold().ReportViewer("viewer")
    .ProcessingMode(BoldReports.ReportViewerEnums.ProcessingMode.Local)
    .ReportPath("~/Resources/product-list.rdlc")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportLoaded("onReportLoaded")
)

<script type="text/javascript">
function onReportLoaded(args) {
    var dataSource = [
        {
            ProductName: "Baked Chicken and Cheese", OrderId: "323B60", Price: 55, Category: "Non-Veg",
            Ingredients: "Grilled chicken, Corn and Olives.", ProductImage: ""
        },
        {
            ProductName: "Chicken Delite", OrderId: "323B61", Price: 100, Category: "Non-Veg", Ingredients:
            "Cheese, Chicken chunks, Onions & Pineapple chunks.", ProductImage: ""
        },
        {
            ProductName: "Chicken Tikka", OrderId: "323B62", Price: 64, Category: "Non-Veg", Ingredients: "Onions,
            Grilled chicken, Chicken salami & Tomatoes.", ProductImage: ""
        }];
    var reportObj = $('#viewer').data("boldReportViewer");
    reportObj.model.dataSources = [
        {
            value: ej.DataManager(dataSource).executeLocal(ej.Query()),
            name: "list"
        }
    ];
}
</script>
`

```

Report error

When an error occurs in the report processing, it raises the **ReportError** event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```

`js
@(Html.Bold().ReportViewer("viewer")

```

```
.ProcessingMode(BoldReports.ReportViewerEnums.ProcessingMode.Local)
.ReportPath("~/Resources/product-list.rdlc")
.ReportServiceUrl("/api/ReportViewer")
.ReportError("onReportError")
)

<script type="text/javascript">
function onReportError(args) {
    alert(args.errmsg);
    args.cancel = true;
}
</script>
`
```

To suppress the default error dialog, set the cancel argument to true.

[Show error](#)

The **showError** event is invoked whenever users click a report item that contains an error in processing. It provides detailed information about the cause of error. You can hide the default dialog and show your customized dialog.

```
`js
@(Html.Bold().ReportViewer("viewer")
.ProcessingMode(BoldReports.ReportViewerEnums.ProcessingMode.Local)
.ReportPath("~/Resources/product-list.rdlc")
.ReportServiceUrl("/api/ReportViewer")
.showError("onShowError")
)

<script type="text/javascript">
function onShowError(args) {
    alert("Error code : " + args.errorCode + "\n" +
    "Error Detail : " + args.errorDetail + "\n" +
    "Error Message : " + args.errorMessage);
    args.cancel = true;
}
</script>
`
```

Drill through

When a drill through item is selected in a report, it invokes the `DrillThrough` event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-person-details.rdl")
    .DrillThrough("onDrillThrough")
)
<script type="text/javascript">
function onDrillThrough(args) {
    args.actionInfo.ReportName = "personal-details";
    args.actionInfo.Parameters = [{ name: 'EmployeeID', value: ['3'] }];
}
</script>
`
```

Hyperlink

The `Hyperlink` event occurs when users click a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-person-details.rdl")
    .Hyperlink("onHyperlink")
)
<script type="text/javascript">
function onHyperlink(args) {
    args.cancel = true;
    //You can modify the URL here
    window.open(args.actionInfo.Hyperlink);
}
</script>
`
```

Handle post actions

Report processing actions are sent in an Ajax request to exchange data with the Web API service. You can handle post actions event to customize the Ajax requests.

AjaxBeforeLoad

AjaxSuccess

AjaxError

AjaxBeforeLoad

This event can be triggered before an Ajax request is sent to the Report Viewer Web API service. It allows you to set additional headers and custom data in the Ajax request. The following code sample demonstrates adding custom authorization header and passing default parameter values to service.

Add custom header to Ajax request

Initialize the AjaxBeforeLoad event in the script and add the authorization token to the headers property.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .AjaxBeforeLoad("onAjaxRequest")
)
<script type="text/javascript">
function onAjaxRequest(args) {
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
}
</script>
`
```

In this tutorial, the sales-order-detail.rdl report is used, and it can be downloaded from [here](#).

Get the custom header value from the HttpContext header collection using the key name specified at client side.

```
'csharp
string authenticationHeader;
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    if (jsonResult != null)
```

```
{  
//Get client side custom ajax header and store in local variable  
authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];  
//Perform your custom validation here  
if (authenticationHeader == "")  
{  
    return new Exception("Authentication failed!!!!");  
}  
else  
{  
    return ReportHelper.ProcessReport(jsonResult, this);  
}  
}  
  
return null;  
}  
`
```

Perform your own action to validate the header values.

Pass custom data in Ajax request

Use the **data** property to set custom data to the server in the Ajax request. In the following code sample, parameter values are passed to the server side.

```
`js  
@(Html.Bold().ReportViewer("viewer")  
.ReportServiceUrl("/api/ReportViewer")  
.ReportPath("~/Resources/sales-order-detail.rdl")  
.AjaxBeforeLoad("onAjaxRequest")  
)  
  
<script type="text/javascript">  
function onAjaxRequest(args) {  
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });  
    //Passing custom parameter data to server  
    args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];  
}  
</script>  
`
```

The custom data values are stored in the `customData` header key, you can store it to the local property. The following code sample stores parameter values, then the values are set to the report in the `OnReportLoaded` method.

'csharp

```
string authenticationHeader;
public string DefaultParameter = null;
public object PostReportAction([FromBody] Dictionary<string, object> jsonResult)
{
    if (jsonResult != null)
    {
        if (jsonResult.ContainsKey("customData"))
        {
            //Get client side custom data and store in local variable. Here parameter values are sent.
            DefaultParameter = jsonResult["customData"].ToString();
        }

        //Get client side custom ajax header and store in local variable
        authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
        //Perform your custom validation here
        if (authenticationHeader == "")
        {
            return new Exception("Authentication failed!!!");
        }
        else
        {
            return ReportHelper.ProcessReport(jsonResult, this);
        }
    }
    return null;
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
    if (DefaultParameter != null)
    {
        //Set client side custom header data
    }
}
```

Handle post actions

AjaxSuccess

```
reportOption.ReportModel.Parameters =
JsonConvert.DeserializeObject<List<BoldReports.Web.ReportParameter>>(DefaultParameter);
}
}
`
```

AjaxSuccess

To perform custom action or show user defined message, use the **AjaxSuccess** event on the successful Ajax request.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .AjaxBeforeLoad("onAjaxRequest")
    .AjaxSuccess("onAjaxSuccess")
)
<script type="text/javascript">
function onAjaxRequest(args) {
    args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
    //Passing custom parameter data to server
    args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
function onAjaxSuccess(args) {
    //Perform your custom success message here
    alert("Ajax request success!!!");
}
</script>
`
```

AjaxError

The **AjaxError** event is called, if an error occurred with the request, you can display the customized error detail in the event method.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
```

```
.AjaxBeforeLoad("onAjaxRequest")
.AjaxError("onAjaxFailure")
)
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
function onAjaxFailure(args) {
alert("Status: " + args.status + "\n" +
"Error: " + args.responseText);
}
</script>
`
```

You can never have both an error and a success callback with a request.

Error logging in ASP.NET MVC Report Viewer

If an error occurred in report processing, ASP.NET MVC Report Viewer displays short messages about the error in view. You need to configure the `ApiController` to save all logs, stack trace, and error information into a physical file location.

This section explains how to log the detailed error information to your ASP.NET MVC application.

This section requires a ASP.NET MVC Report Viewer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

In Solution Explorer, open the Report Viewer Controller file.

Inherit the `IReportLogger` interface and implement the interface methods.

```
'csharp
public class ReportViewerController : ApiController, IReportController, IReportLogger
{
    public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
    {
        throw new NotImplementedException();
    }
}
```

```
public void LogError(string errorCode, string message, Exception exception, string errorDetail, string  
methodName, string className)  
{  
    throw new NotImplementedException();  
}  
}  
`
```

Create a method in `ReportViewerController` to write the error text into application folder.

```
`csharp  
internal void WriteLogs(string errorMessage)  
{  
    string filePath = HttpContext.Current.Server.MapPath("/App_Data/Errordetails.txt");  
    using (StreamWriter writer = new StreamWriter(filePath, true))  
    {  
        writer.AutoFlush = true;  
        writer.WriteLine(errorMessage);  
    }  
}
```

Invoke the newly created function in `.LogError` as follows.

```
`csharp  
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType  
errorType)  
{  
    WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));  
}  
public void LogError(string errorCode, string message, Exception exception, string errorDetail, string  
methodName, string className)  
{  
    WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:  
{3}", className, methodName, errorDetail, exception.StackTrace));  
}
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final controller is given as follows, you can replace it in your application.

```
'csharp
public class ReportViewerController : ApiController, IReportController, IReportLogger
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]
    public object GetResource(string key, string resourcetype, bool isPrint)
    {
        return ReportHelper.GetResource(key, resourcetype, isPrint);
    }

    // Method that will be called when initialize the report options before start processing the report
    public void OnInitReportOptions(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }

    // Method that will be called when reported is loaded
    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }

    public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
    {
        WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
    }
}
```

```
public void LogError(string errorCode, string message, Exception exception, string errorDetail, string  
methodName, string className)  
{  
    WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:  
{3}", className, methodName, errorDetail, exception.StackTrace));  
}  
  
internal void WriteLogs(string errorMessage)  
{  
    // Error details text file path location  
    string filePath = HttpContext.Current.Server.MapPath("/App_Data/Errordetails.txt");  
    using (StreamWriter writer = new StreamWriter(filePath, true))  
    {  
        writer.AutoFlush = true;  
        writer.WriteLine(errorMessage);  
    }  
}  
}  
}  
`
```

Print report

The Report Viewer provides print report option in the toolbar to print a copy of the report. The page setup dialog allows you to set the paper size or other page setup properties. To see print margins, click **Print Layout** on the toolbar.

You can set values in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values of the Page Setup dialog is based on the report properties set in the design view.

[View report in print mode](#)

Print margins are displayed in the print layout only. To view report in print mode by default, set the **PrintMode** property to true.

```
`js  
@(Html.Bold().ReportViewer("viewer")  
.ReportServiceUrl("/api/ReportViewer")  
.ReportPath("~/Resources/sales-order-detail.rdl")  
.PrintMode(true)  
)  
`
```

By default, the Report Viewer renders report in normal layout in which the print margins are not displayed.

Print in new page

To open the print in a new tab of the current browser, set the `printOption` property to `NewTab`. By default, it shows the print dialog in the same page.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .PrintMode(true)
    .PrintOption(BoldReports.ReportViewerEnums.PrintOptions.NewTab)
)
'
```

The pop-up blocker should be enabled for the page to open the print view in new tab.

Set page orientation and paper size

You can specify the print page paper size and orientation at client-side to change the page setup properties by setting the `PageSettings` property.

The following code example illustrates how to set page orientation and paper size in the Report Viewer for client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .PrintMode(true)
    .PageSettings(page =>
        page.Orientation(BoldReports.ReportViewerEnums.Orientation.Landscape).PaperSize(BoldReports.ReportViewerEnums.PaperSize.Letter))
)
'
```

Set report margin

To set margin values to the report page setup, use the `Margins` property and specify the value to top, right, bottom, and left.

The following code example illustrates how to set report margin in the Report Viewer for client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
```

Print report

Set page height and width

```
.ReportPath("~/Resources/sales-order-detail.rdl")
.PrintMode(true)
.PageSettings(page => page.Margins(margin => margin.Top(0.5).Left(0.5).Bottom(0.5).Right(0.5)))
)
```

The values set in the margin property is considered as inches input.

Set page height and width

To set height and width values to the report page setup, use the **Height** and **Width** properties.

The following code example illustrates how to set page height and width in the Report Viewer for client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
.ReportServiceUrl("/api/ReportViewer")
.ReportPath("~/Resources/sales-order-detail.rdl")
.PrintMode(true)
.PageSettings(page => page.Height(11.69).Width(8.27))
)
```

The values set in the height and width properties are considered as inches input.

Print report with images

When the report has more images, the browser will send the report stream to the print dialog before the images are completely loaded. To load the print report stream with complete images, you should set the **EmbedImageData** property to true in **OnInitReportOptions** as shown in the following code.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.EmbedImageData = true;
}
```

Replace the following code sample in the client-side.

```
'js
@(Html.Bold().ReportViewer("viewer")
.ReportServiceUrl("/api/PrintWithImages")
.ReportPath("~/Resources/product-details.rdl")
```

```
)  
`
```

In this tutorial, the `product-details.rdl` report is used, and it can be downloaded from [here](#).

External styles in report printing

While printing report, the external styles are used in the application overrides printable page style and prints output with incorrect alignments. To avoid the external script overriding, set the `isStyleLoad` property to false, which will print the page using only the Report Viewer styles.

```
'js  
@(Html.Bold().ReportViewer("viewer")  
.ReportServiceUrl("/api/PrintWithImages")  
.ReportPath("~/Resources/product-details.rdl")  
.ReportPrint("onReportPrint")  
)  
<script type="text/javascript">  
function onReportPrint(args) {  
args.isStyleLoad = false;  
}  
</script>  
`
```

Show print progress

Report Viewer provides events to show the progress information when the printing takes long time to complete.

To show print progress, follow these steps:

Set the `PrintProgressChanged` event in Report Viewer initialization.

Implement the function and add code samples to show a custom message based on the print progress status as shown in the following code snippet.

```
'js  
@(Html.Bold().ReportViewer("viewer")  
.ReportServiceUrl("/api/PrintWithImages")  
.ReportPath("~/Resources/product-details.rdl")  
.PrintProgressChanged("onPrintProgressChanged")  
)  
<script type="text/javascript">
```

Print report

Remove empty spaces in printing

```
function onPrintProgressChanged(args) {  
    if (args.stage == "beginPrint") {  
        $('#viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing print data.. Please wait..." });  
    }  
    if (args.stage == "printStarted") {  
        var popupObj = $('#viewer').data('ejWaitingPopup');  
        popupObj.hide();  
    }  
    else if (args.stage == "preparation") {  
        console.log(args.stage);  
        if (args.preparationStage == "dataPreparation") {  
            console.log(args.preparationStage);  
            console.log(args.totalPages);  
            console.log(args.currentPage);  
            if (args.totalPages > 1 && args.currentPage > 1) {  
                var progressPercentage = Math.floor((args.currentPage / args.totalPages) * 100);  
                if (progressPercentage > 0) {  
                    var popupObj = $('#viewer').data('ejWaitingPopup');  
                    popupObj.setModel({ text: "Preparing print data.." + progressPercentage + " % completed.. Please wait..." });  
                }  
            }  
        }  
        args.handled = true;  
    }  
}
```

`

Remove empty spaces in printing

The extra blank page is created when the body of your report is too wide for your page. To make the report appear on a single page, all the content within the report body must fit on the physical page, and the body width should be as the following formula.

Body Width <= Page Width - (Left Margin + Right Margin)

For more details about removing the empty pages in the report while designing, refer to the knowledge base article of [report page sizing](#).

Export report

The Report Viewer provides events and properties to control and customize the report exporting functionality.

Export event handling

You can show the progress information, when the exporting process takes long time to complete using the `ExportProgressChanged` event.

Set the `ExportProgressChanged` event in Report Viewer initialization.

Implement the function and replace the following code samples to show a custom message based on the progress stage.

The following code example demonstrates how to export event handling in the Report Viewer at client side.

```
 `js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ExportProgressChanged("onExportProgressChanged")
)
<script type="text/javascript">
function onExportProgressChanged(args) {
    if (args.stage === "beginExport") {
        console.log(args.stage);
        args.format =
            $('#viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing exporting document.. Please wait..." });
    }
    else if (args.stage === "exportStarted") {
        console.log(args.stage);
        var popupObj1 = $('#viewer').data('ejWaitingPopup');
        popupObj1.hide();
    }
    else if (args.stage === "preparation") {
        console.log(args.stage);
    }
}
```

```
console.log(args.format);
console.log(args.preparationStage);
if (args.format === "PDF" && args.preparationStage === "documentPreparation") {
    console.log(args.totalPages);
    console.log(args.currentPage);
    if (args.totalPages > 1 && args.currentPage > 1) {
        var progressPercentage = Math.floor((args.currentPage / args.totalPages) * 100);
        if (progressPercentage > 0) {
            var popupObj2 = $('#viewer').data('ejWaitingPopup');
            popupObj2.setModel({ text: "Preparing exporting document.." + progressPercentage + " % completed..
Please wait..." });
        }
    }
}
args.handled = true;
}
</script>
`
```

Export data visualization items

To export the reports with data visualization components, it is mandatory to configure the web scripts in Report Viewer Web API controller. If the report definition uses chart, gauge and map report items then configure the scripts in Web API as the following steps,

Open the Report Viewer Web API controller.

Configure the below scripts and styles in **OnInitReportOptions** on Web API controller.

jquery-1.10.2.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js

ej2-base.min.js

ej2-data.min.js

```
ej2-pdf-export.min.js  
ej2-svg-base.min.js  
ej2-lineargauge.min.js  
ej2-circulargauge.min.js  
ej.map.min.js  
bold.report-viewer.min.js
```

You can replace the following codes in Report Viewer Web API controller.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>  
    {  
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",  
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",  
        //Chart component script  
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",  
        //Gauge component scripts  
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",  
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",  
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",  
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",  
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",  
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",  
        //Map component script  
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",  
        //Report Viewer Script  
        "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",  
    };  
    reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>  
    {
```

```
"https://code.jquery.com/jquery-1.10.2.min.js"
};

}

`
```

The data visualization components will not export without the above script configurations.

Export data visualization items in azure environment

Report Viewer uses **WebBrowser** to export the data visualization items to PDF, Word, Excel file formats. The **WebBrowser** is not supported in azure environment. To overcome this limitation in azure environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

To download **PhantomJS** application and deploy it on your machine, you should accept it's license terms on [LICENSE](#) and [Third-Party](#) document.

Download **PhantomJS** from [here](#) and extract the download file.

Copy the **PhantomJS.exe** file from the extracted bin folder and paste into **PhantomJS** folder in your application.

Open the Report Viewer Web API controller.

Set the **UsePhantomJS** property to true and **PhantomJSPath** property in **OnInitReportOptions** method.

```
'csharp
// Method will be called to initialize the report information to load the report with ReportHelper for
processing.

public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ExportResources.UsePhantomJS = true;
    reportOption.ReportModel.ExportResources.PhantomJSPath = @"\PhantomJS\";
    reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
    {
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
        "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
        //Chart component script
        "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
        //Gauge component scripts
        "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
    }
}
```

```

"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
//Map component script
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
//Report Viewer Script
"https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
};

reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
  "https://code.jquery.com/jquery-1.10.2.min.js"
};
}
`
```

The **Scripts** and **Dependent** scripts must be added to export the items. For more details refer to the [export data visualization items](#) section.

Change Excel and Word export format

Allows you change the default file format to any other file format using the **ExcelFormats** and **WordFormats** properties.

The following code example demonstrates how to change Excel and Word export format in the Report Viewer at client side.

```

js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ExportSettings(export =>
        export.ExcelVersion(BoldReports.ReportViewerEnums.ExcelFormats.Excel2013).WordFormatType(BoldReports.ReportViewerEnums.WordFormats.Word2013)
    )
)`
```

Hide specific export type for report

Show or hide the default export types available in the component using the **ExportOptions** property.

The following code example demonstrates how to hide specific export type to the report in the Report Viewer at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ExportSettings(export => export.ExportOptions(BoldReports.ReportViewerEnums.ExportOptions.All &
        ~BoldReports.ReportViewerEnums.ExportOptions.PDF)))
)
```

PDF export options

The **PDFOptions** provides properties to manage PDF export behaviors. You should set the properties in the **OnInitReportOptions** method of the Web API service.

Export with complex scripts

To export reports with the complex scripts, set the **ComplexScript** property of **PDFOptions** instance to true.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
    {
        EnableComplexScript = true
    };
}
```

PDF conformance

You can export the report as a **PDF/A-1b** document by specifying the **PdfConformanceLevel.Pdf_A1B** conformance level in the **PdfConformanceLevel** property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
    {
        PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B
    };
}
```

```
}
```

Add custom PDF fonts

You can add custom fonts to the PDF exported document by adding the font streams to **Fonts** collection in **PDFOptions** instance.

To add custom fonts to the PDF exported document, follow these steps:

Add the font **.ttf** files into your application **Resources** folder.

In the Solution Explorer, open the properties of the font file and set the **Copy** property to **Output Directory** as **Copy always**.

Initialize the **Font** collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    string basePath = _hostingEnvironment.WebRootPath;
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
    {
        //Load Missing font stream
        Fonts = new Dictionary<string, System.IO.Stream>
        {
            { "Segoe UI", new FileStream(basePath + @"\Resources\font_symbols.ttf", FileMode.Open,
                FileAccess.Read) }
        }
    };
}
```

If any fonts used in the report definition is not installed or available in the local system, then you should load the font stream. In the above code, **font_symbols** font stream is loaded to export the **sales-order-detail.rdl** report as symbols.

Word export options

The **WordOptions** provides properties to manage Word document export behaviors.

Word document type

You can save the report to the required document version by setting the **FormatType** property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
    {
        FormatType = BoldReports.Writer.WordFormatType.Docx
    };
}
`
```

Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the `LayoutOption` to `TopLevel`. The `ParagraphSpacing` is the distance value added between two elements in the document.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
    {
        LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel,
        ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()
        {
            Bottom = 0.5f,
            Top = 0.5f
        }
    };
}
`
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, add an empty paragraph between two tables.

Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without password. The following are the types of protection:

AllowOnlyComments: Adds or modifies only the comments in the Word document.

AllowOnlyFormFields: Modifies the form field values in the Word document.

AllowOnlyRevisions: Accepts or rejects the revisions in the Word document.

AllowOnlyReading: Views the content only in the Word document.

NoProtection: Accesses or edits the Word document contents as normally.

`csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
    {
        ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading
    };
}
```

Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

`csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
    {
        ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013
    };
}
```

Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the **LayoutOption** to **IgnoreCellMerge**.

`csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
```

```
{  
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge  
};  
}  
'
```

Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the `ExcelSheetProtection` or enabling the `ReadOnlyRecommended` properties.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
{  
ReadOnlyRecommended = true,  
ExcelSheetProtection = Syncfusion.XlsIO.ExcelSheetProtection.DeletingColumns  
};  
}  
'
```

CSV export options

The `CsvOptions` allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
reportOption.ReportModel.CsvOptions = new BoldReports.Writer.CsvOptions()  
{  
Encoding = System.Text.Encoding.Default,  
FieldDelimiter = ",",  
UseFormattedValues = false,  
Qualifier = "#",  
RecordDelimiter = "@",  
SuppressLineBreaks = true,
```

Export report

Password protect exported document

```
FileExtension = ".txt"  
};  
}  
`
```

Password protect exported document

Allows you protect the exported document such as PDF, Word, Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet illustrates how to encrypt the exported document with user-defined password.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    //PDF encryption  
    reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions();  
    reportOption.ReportModel.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()  
    {  
        UserPassword = "password"  
    };  
    //Word encryption  
    reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()  
    {  
        EncryptionPassword = "password"  
    };  
    //Excel encryption  
    reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()  
    {  
        PasswordToModify = "password",  
        PasswordToOpen = "password"  
    };  
    //PPT encryption  
    reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()  
    {  
        EncryptionPassword = "password"  
    };  
}
```

>Password protection is not supported for HTML export format.

Custom Actions

Add user defined buttons to the toolbar and invoke custom actions using the **Report Viewer** property. You can create a custom email button to send an email with the rendered report to users.

Add email button

Create an email button in the toolbar using the **CustomItems** property with the values such as **GroupIndex**, **Index**, **Type**, **CssClass**, and **Tooltip**. The **ToolBarItemClick** event triggers when you click the email button.

Access the Report Viewer model and create a JSON array for sending requests to the Web API Server. You can use the following codes to create an event with custom action.

You can use the following codes to add email button from controller and passing the data to view using **ViewBag**.

```
'csharp
public ActionResult Index()
{
    ToolbarSettings toolbarSettings = new ToolbarSettings();
    toolbarSettings.CustomItems = new List<CustomItem>();
    var customItem = new CustomItem()
    {
        GroupIndex = 1,
        Index = 1,
        CssClass = "e-icon e-mail e-reportviewer-icon",
        Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
        Id = "E-Mail",
        Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
    };
    toolbarSettings.CustomItems.Add(customItem);
    ViewBag.toolbarSettings = toolbarSettings;
    return View();
}
```

You can use the following codes to set an **ToolbarSettings** property at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbarSettings =>
        toolbarSettings.CustomItems(@ViewBag.toolbarSettings.CustomItems))
)
`
```

You can use the following codes to create an **ToolBarItemClick** event at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolBarItemClick("onToolBarItemClick"))
)
<script type="text/javascript">
function onToolBarItemClick(args) {
    alert('Action Triggered');
}
</script>
`
```

You can use the following codes to get **ToolbarSettings** properties on a dynamic object using **ViewBag.toolbarSettings.CustomItems** and invoke custom actions using custom email button at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbarSettings =>
        toolbarSettings.CustomItems(@ViewBag.toolbarSettings.CustomItems))
    .ToolBarItemClick("onToolBarItemClick")
)
`
```

```
)  
<script type="text/javascript">  
function onToolBarItemClick(args) {  
if (args.value == "E-Mail") {  
var proxy = $('#viewer').data('boldReportViewer');  
var Report = proxy.model.reportPath;  
var lastsIndex = Report.lastIndexOf("/");  
var reportName = Report.substring(lastsIndex + 1);  
var requrl = proxy.model.reportServiceUrl + '/SendEmail';  
var _json = {  
exportType: "PDF", reportViewerToken: proxy._reportViewerToken, ReportName: reportName  
};  
$.ajax({  
type: "POST",  
contentType: "application/json; charset=utf-8",  
url: requrl,  
data: JSON.stringify(_json),  
dataType: "json",  
crossDomain: true  
})  
}  
}  
  
</script>  
'
```

Create custom email action

Create a new action method **SendEmail** in the Web API service.

Export the report to the required type using the **ReportHelper.GetReport** method to send a report stream as an attachment.

The following code sample exports the report to stream and send it as an attachment to a specified mail address. In the code, the **SmtpClient** method is used to send the report as an email attachment.

`csharp

```
public object SendEmail(Dictionary<string, object> jsonResult)
{
    string _token = jsonResult["reportViewerToken"].ToString();
    var stream = ReportHelper.GetReport(_token, jsonResult["exportType"].ToString());
    stream.Position = 0;
    if (!ComposeEmail(stream, jsonResult["reportName"].ToString()))
    {
        return "Mail not sent !!!";
    }
    return "Mail Sent !!!";
}

public bool ComposeEmail(Stream stream, string reportName)
{
    try
    {
        MailMessage mail = new MailMessage();
        SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");
        mail.IsBodyHtml = true;
        mail.From = new MailAddress("xx@gmail.com");
        mail.To.Add("xx@gmail.com");
        mail.Subject = "Report Name : " + reportName;
        stream.Position = 0;
        if (stream != null)
        {
            ContentType ct = new ContentType();
            ct.Name = reportName + DateTime.Now.ToString() + ".pdf";
            System.Net.Mail.Attachment attachment = new System.Net.Mail.Attachment(stream, ct);
            mail.Attachments.Add(attachment);
        }
        SmtpServer.Port = 587;
        SmtpServer.Credentials = new System.Net.NetworkCredential("xx@gmail.com", "xx");
        SmtpServer.EnableSsl = true;
        SmtpServer.Send(mail);
    }
}
```

```
return true;  
}  
  
catch (Exception ex)  
{  
  
    throw ex;  
}  
  
return false;  
}  
  
'
```

In the above code sample, the report is exported to PDF format and send to users using the `SmptClient` method.

Toolbar customization

You can hide the component toolbar to show customized user interface or to customize the toolbar icons and element's appearances using the templates and Report Viewer toolbar customization properties.

In this tutorial, the `sales-order-detail.rdl` report is used and it can be downloaded from [here](#). You can add the reports from the Syncfusion installation location. For more information, refer to [Samples and demos](#).

Hide parameter block and toolbar items

To hide toolbar items, set the `ToolbarSettings` property. The following code can be used to remove the parameter option from the toolbar and hide the parameter block.

The following code example demonstrates how to hide the parameter block in the Report Viewer at client side.

```
'js  
  
@(Html.Bold().ReportViewer("viewer")  
    .ReportServiceUrl("/api/ReportViewer")  
    .ReportPath("~/Resources/sales-order-detail.rdl")  
    .ToolbarSettings(toolbar => toolbar.Items(BoldReports.ReportViewerEnums.ToolbarItems.All &  
        ~BoldReports.ReportViewerEnums.ToolbarItems.Parameters))  
)
```

Similarly, you can show or hide all other toolbar options with the help of `ToolbarSettings.Items` enum.

Enable stop option in toolbar

To enable stop option in toolbar, set the `ToolbarSettings.Items` property to `BoldReports.ReportViewerEnums.ToolbarItems.All`. The following code can be used to enable stop option in toolbar.

The following code example demonstrates how to enable stop option in the Report Viewer toolbar at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbar => toolbar.Items(BoldReports.ReportViewerEnums.ToolbarItems.All))
)
`
```

Hide toolbar

To hide the Report Viewer toolbar, set the **ShowToolbar** property to false.

The following code example demonstrates how to hide the toolbar in the Report Viewer at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbar => toolbar.ShowToolbar(false))
)
`
```

Decide or hide the export option

The Report Viewer provides the **ExportOptions** property to show or hide the default export types available in the component. The following code hides the HTML export type from the default export options.

The following code example demonstrates how to decide or hide the export option in the Report Viewer at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ExportSettings(export => export.ExportOptions(BoldReports.ReportViewerEnums.ExportOptions.All &
        ~BoldReports.ReportViewerEnums.ExportOptions.Html))
)
`
```

Add custom items to the export drop-down

To add custom items to the export drop-down available in the Report Viewer toolbar, use the property `CustomItems` and provide the JSON array of collection input with the `Index`, `CssClass` name, and `Value` properties. Register the `ExportItemClick` event to handle the custom item actions as given in following code snippet.

You can use the following codes to add custom items to the export drop-down from controller and passing the data to view using `ViewBag`.

```
'csharp
public ActionResult Index()
{
    ExportSettings exportSettings = new ExportSettings();
    exportSettings.CustomItems = new List<CustomExportItem>();
    var exportItem1 = new CustomExportItem() { Index = 2, CssClass = "", Value = "Text File" };
    var exportItem2 = new CustomExportItem() { Index = 4, CssClass = "", Value = "DOT" };
    exportSettings.CustomItems.Add(exportItem1);
    exportSettings.CustomItems.Add(exportItem2);
    ViewBag.exportSettings = exportSettings;
    return View();
}
`
```

You can use the following codes to set an `ExportSettings` property at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ExportSettings(exportSettings => exportSettings.CustomItems(@ViewBag.exportSettings.CustomItems))
)
`
```

You can use the following codes to create an `ExportItemClick` event at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ExportItemClick("onExportItemClick")
)
`
```

```
)  
<script type="text/javascript">  
function onExportItemClick(args) {  
    alert('Action Triggered');  
}  
</script>  
\
```

You can use the following codes to get `ExportSettings` properties on a dynamic object using `ViewBag.exportSettings.CustomItems` and invoke custom actions.

```
'js  
@(Html.Bold().ReportViewer("viewer")  
.ReportServiceUrl("/api/ReportViewer")  
.ReportPath("~/Resources/sales-order-detail.rdl")  
.ExportSettings(exportSettings =>  
    exportSettings.CustomItems(@ViewBag.exportSettings.CustomItems)).ExportItemClick("onExportItemClick")  
)  
<script type="text/javascript">  
//Export click event handler  
function onExportItemClick(args) {  
    if (args.value === "Text File") {  
        //Implement the code to export report as Text  
        alert("Text File export option clicked");  
    } else if (args.value === "DOT") {  
        //Implement the code to export report as DOT  
        alert("DOT export option clicked");  
    }  
}  
</script>  
\
```

Add custom toolbar item

You can add custom items to Report Viewer toolbar using the `ToolbarSettings` property. You must register the `ToolBarItemClick` event to handle the newly added custom items action.

You can use the following codes to add custom toolbar item from controller and passing the data to view using `ViewBag`.

```
'csharp
public ActionResult Index()
{
    ToolbarSettings toolbarSettings = new ToolbarSettings();
    toolbarSettings.CustomItems = new List<CustomItem>();
    var customItem = new CustomItem()
    {
        GroupIndex = 1,
        Index = 1,
        CssClass = "e-icon e-mail e-reportviewer-icon",
        Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
        Id = "E-Mail",
        Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
    };
    toolbarSettings.CustomItems.Add(customItem);
    ViewBag.toolbarSettings = toolbarSettings;
    return View();
}
`
```

You can use the following codes to set an `ToolbarSettings` property at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbarSettings =>
    toolbarSettings.CustomItems(@ViewBag.toolbarSettings.CustomItems))
)
`
```

You can use the following codes to create an `ToolBarItemClick` event at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
```

```

.ToolBarItemClick("onToolBarItemClick")
)
<script type="text/javascript">
function onToolBarItemClick(args) {
alert('Action Triggered');
}
</script>
`
```

You can use the following codes to get `ToolbarSettings` properties on a dynamic object using `ViewBag.toolbarSettings.CustomItems` and invoke custom actions.

```

`js
@(Html.Bold().ReportViewer("viewer")
.ReportServiceUrl("/api/ReportViewer")
.ReportPath("~/Resources/sales-order-detail.rdl")
.ToolbarSettings(toolbarSettings =>
toolbarSettings.CustomItems(@ViewBag.toolbarSettings.CustomItems))
.ToolBarItemClick("onToolBarItemClick")
)
<script type="text/javascript">
function onToolBarItemClick(args) {
alert('Action Triggered');
}
</script>
`
```

Add custom item to exiting toolbar group

To add a custom item to existing toolbar group use the property `CustomGroups` in `ToolbarSettings` and provide the JSON array of collection input with the `GroupIndex`, `Items`, `Type`, `CssClass` `Id`, and `Tooltip` properties as given in following code snippet.

You can use the following codes to add custom item to exiting toolbar group from controller and passing the data to view using `ViewBag`.

```

`csharp
public ActionResult Index()
{
    ToolbarSettings toolbarSettings = new ToolbarSettings();
    var groupItem = new List<CustomItem>();
```

```
groupItem.Add(new CustomItem()
{
    CssClass = "e-icon e-mail e-reportviewer-icon CustomGroup",
    Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
    Id = "CustomGroup",
    Tooltip = new ToolTip() { Header = "CustomGroup", Content = "toolbargroups" }
});
groupItem.Add(new CustomItem()
{
    CssClass = "e-icon e-mail e-reportviewer-icon subCustomGroup",
    Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
    Id = "subCustomGroup",
    Tooltip = new ToolTip() { Header = "subCustomGroup", Content = "subtoolbargroups" }
});
toolbarSettings.CustomGroups.Add(new CustomGroup() { Items = groupItem, GroupIndex = 4 });
ViewBag.toolbarSettings = toolbarSettings;
return View();
}
`
```

You can use the following codes to set an `ToolbarSettings` property at client side.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbarSettings =>
        toolbarSettings.CustomGroups(@ViewBag.toolbarSettings.CustomGroups)
    )
)`
```

You can use the following codes to create an `ToolBarItemClick` event at client side.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
```

```
.ToolBarItemClick("onToolBarItemClick")
)
<script type="text/javascript">
function onToolBarItemClick(args) {
alert('Action Triggered');
}
</script>
`
```

You can use the following codes to get `ToolbarSettings` properties on a dynamic object using `ViewBag.toolbarSettings.CustomGroups` and invoke custom actions.

```
`js
@(Html.Bold().ReportViewer("viewer")
.ReportServiceUrl("/api/ReportViewer")
.ReportPath("~/Resources/sales-order-detail.rdl")
.ToolbarSettings(toolbarSettings =>
toolbarSettings.CustomGroups(@ViewBag.toolbarSettings.CustomGroups))
.ToolBarItemClick("onToolBarItemClick")
)
<script type="text/javascript">
function onToolBarItemClick(args) {
if (args.value === "CustomGroup") {
//Implement the code to CustomGroup toolbar option
alert("CustomGroup toolbar option clicked");
}
if (args.value === "subCustomGroup") {
//Implement the code to subCustomGroup toolbar option
alert("SubCustomGroup toolbar option clicked");
}
}
</script>
`
```

Add new toolbar group

To add new toolbar group and custom items to it, use the property `CustomItems` in `ToolbarSettings` and provide the JSON array of collection input with the `GroupIndex`, `Index` properties. The `CustomItem` must have the properties `Type`, `CssClass` and `Tooltip` as given in following code snippet.

You can use the following codes to add new toolbar group from controller and passing the data to view using `ViewBag`.

```
'csharp
public ActionResult Index()
{
    ToolbarSettings toolbarSettings = new ToolbarSettings();
    toolbarSettings.CustomItems = new List<CustomItem>();
    var customItem = new CustomItem()
    {
        GroupIndex = 1,
        Index = 1,
        CssClass = "e-icon e-mail e-reportviewer-icon",
        Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
        Id = "E-Mail",
        Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
    };
    toolbarSettings.CustomItems.Add(customItem);
    ViewBag.toolbarSettings = toolbarSettings;
    return View();
}
'
```

You can use the following codes to set an `ToolbarSettings` property at client side.

```
'js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .ToolbarSettings(toolbarSettings =>
        toolbarSettings.CustomItems(@ViewBag.toolbarSettings.CustomItems))
)
'
```

Localization of Bold Reports ASP.NET MVC Report Viewer

Localization of ASP.NET MVC Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. To render the static text with specific culture, refer to the following corresponding culture script files and set culture name to the `locale` property of the Report Viewer.

`ej.localetexts.fr-FR.min.js`

`ej.culture.fr-FR.min.js`

Install the `BoldReports.Global` Nuget package in your MVC application.

Refer to this `ej.localetexts.fr-FR.min.js` script file from the `Scripts` folder of your application.

`html

```
<script src="~/Scripts/bold-reports/i10n/reportdesigner/ej.localetexts.fr-FR.min.js"></script>
```

`

Refer to this `ej.culture.fr-FR.min.js` script file from the `Scripts` folder of your application.

`html

```
<script src="~/Scripts/bold-reports/i18n/ej.culture.fr-FR.min.js"></script>
```

`

To render the localization Report Viewer, use the following code snippet.

The following code example illustrates how to localize Report Viewer in the `Index.cshtml` page.

`js

```
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .Locale("fr-FR")
)
```

`

The following code example illustrates how to localize Report Viewer in the `~/Views/Shared/_Layout.cshtml` page.

`js

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>@ViewBag.Title - Render Report Viewer in French localization</title>
@Styles.Render("~/Content/css")
@Styles.Render("~/Content/material/bold.reports.all.min.css")
@Scripts.Render("~/bundles/modernizr")
</head>
<body>
<div style="min-height: 600px; width: 100%;">
@RenderBody()
</div>
@Scripts.Render("~/bundles/jquery")
@Scripts.Render("~/bundles/bootstrap")
@Scripts.Render("~/Scripts/common/bold.reports.common.min.js")
@Scripts.Render("~/Scripts/common/bold.reports.widgets.min.js")
<!--Renders the chart item. Add this script, only if your report contains the chart report item.-->
@Scripts.Render("~/Scripts/data-visualization/ej.chart.min.js")
<!--Render the gauge item. Add these scripts only if your report contains the gauge report item.-->
@Scripts.Render("~/Scripts/bold-reports/common/ej2-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-data.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-pdf-export.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-svg-base.min.js")
@Scripts.Render("~/Scripts/data-visualization/ej2.circulargauge.min.js")
@Scripts.Render("~/Scripts/data-visualization/ej2.lineargauge.min.js")
<!--Renders the map item. Add this script only if your report contains the map report item.-->
@Scripts.Render("~/Scripts/data-visualization/ej.map.min.js")
<!-- Report Viewer component script-->
@Scripts.Render("~/Scripts/bold.report-viewer.min.js")
<!-- Report Viewer french localization scripts-->
@Scripts.Render("~/Scripts/l10n/ej.localetexts.fr-FR.min.js")
```

```
@Scripts.Render("~/Scripts/i18n/ej.culture.fr-FR.min.js")
@RenderSection("scripts", required: false)
@Html.Bold().ScriptManager()
</body>
</html>
`
```

Responsive layout rendering of ASP.NET MVC Report Viewer

Report Viewer will adaptively render itself with optimal user interfaces for phone, tablet, or desktop form factors. This helps your application to scale elegantly on all form factors with ease. You can enable responsive layout rendering in Report Viewer by setting `isResponsive` property to true.

```
`js
@(Html.Bold().ReportViewer("viewer")
    .ReportServiceUrl("/api/ReportViewer")
    .ReportPath("~/Resources/sales-order-detail.rdl")
    .IsResponsive(true)
)`
```

Normal layout

The following output shows the normal layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in normal layout](/static/assets/aspnet-mvc/report-viewer/images/responsive-layout/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/aspnet-mvc/report-viewer/images/responsive-layout/responsive-layout.png)

Limitations

RDL specification

The Report Viewer control does not support RDL specification for SQL Server 2000 and SQL Server 2005.

Report layout

Vertical alignment in the text box report item is not supported in web rendering.

In the Tablix cell split layout process, the entire cell moves to the next page to display the complete cell items, when the table cell width value exceeds the page width.

Expressions

The object function and VB function do not have complete support.

SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you should specify the data source credentials for each report data source to establish database connection.

Samples and Demos

Browse and explore the ready-to-use RDL, RDLC reports, samples, online, and offline demos.

Locally installed reports

You can obtain the sample RDL and RDLC files from the Bold Reports installed location

`%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.`

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view the ASP.NET MVC Report Viewer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view the GitHub Report Viewer demo samples.

Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of ASP.NET MVC Report Viewer application:

Client-side migration

In the Solution Explorer, right-click the **References** and remove the following assembly references:

`Syncfusion.EJ`

`Syncfusion.EJ.MVC`

Add the assembly from the Syncfusion NuGet package `BoldReports.Mvc5`. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for `BoldReports.Mvc5` NuGet package, and install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Scripts and CSS references

Remove the following scripts and CSS references from the Report Viewer

\Views\Shared_Layout.cshtml page:

ej.web.all.min.css

ej.web.all.min.js

ej.unobtrusive.min.js

The Report Viewer scripts and style sheets are added to the Scripts and Content folders in your application when installing the BoldReports.Mvc5 NuGet package. Add scripts as in the following code sample.

```
'js
@Styles.Render("~/Content/bold-reports/material/bold.reports.all.min.css")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.common.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.widgets.min.js")
<!--Renders the chart item. Add this script, only if your report contains the chart report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.chart.min.js")
<!--Render the gauge item. Add these scripts only if your report contains the gauge report item.-->
@Scripts.Render("~/Scripts/bold-reports/common/ej2-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-data.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-pdf-export.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-svg-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js")
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js")
<!--Renders the map item. Add this script only if your report contains the map report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.map.min.js")
<!-- Report Viewer component script-->
@Scripts.Render("~/Scripts/bold-reports/bold.report-viewer.min.js")
`
```

Open the the ~/Views/Shared/_Layout.cshtml page and replace the script manager tag at the end of element from @Html.EJ().ScriptManager() to @Html.Bold().ScriptManager() as in the following code sample.

'html

```
<body>
...
...
<!-- Bold Report Viewer ScriptManager -->
@Html.Bold().ScriptManager()
</body>
`
```

Adding data visualization scripts

To render the report with data visualization components such as chart, gauge, and map items, add scripts of the visualization element. The following table shows the script reference that needs to be added in Report Viewer page for data visualization elements.

Visualization item | Script file

Chart | ej.chart.min.js|

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item, add chart control script ej.chart.min.js before the bold.report-viewer.min.js reference in the \Views\Shared_Layout.cshtml page as demonstrated in the following code sample.

```
`js
@Styles.Render("~/Content/bold-reports/material/bold.reports.all.min.css")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.common.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.widgets.min.js")
<!--Renders the chart item. Add this script, only if your report contains the chart report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.chart.min.js")
<!-- Report Viewer component script-->
@Scripts.Render("~/Scripts/bold-reports/bold.report-viewer.min.js")
`
```

The following code can be used to render the chart, gauge, and map report items in Report Viewer.

```
`js
@Styles.Render("~/Content/bold-reports/material/bold.reports.all.min.css")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.common.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.widgets.min.js")
<!--Renders the chart item. Add this script, only if your report contains the chart report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.chart.min.js")
```

```
<!--Render the gauge item. Add these scripts only if your report contains the gauge report item.-->
@Scripts.Render("~/Scripts/bold-reports/common/ej2-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-data.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-pdf-export.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-svg-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js")
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js")
<!--Renders the map item. Add this script only if your report contains the map report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.map.min.js")
<!-- Report Viewer component script-->
@Scripts.Render("~/Scripts/bold-reports/bold.report-viewer.min.js")
`
```

Control initialization

The component prefix has been changed from EJ() to SF().

Open the ~/Views/Web.config file and add the BoldReports.Mvc assembly reference to the element.

```
'html
<configuration>
....
....
<system.web.webPages.razor>
....
....
<pages pageBaseType="System.Web.Mvc.WebViewPage">
<namespaces>
....
....
<add namespace="BoldReports.Mvc"/>
</namespaces>
</pages>
</system.web.webPages.razor>
....
```

....
</configuration>
\

Set the value of **UnobtrusiveJavaScriptEnabled** to **false** in Root directory **web.config** file as demonstrated in the following code example.

```
'js  
<configuration>  
<appSettings>  
.....  
.....  
<add key="UnobtrusiveJavaScriptEnabled" value="false" />  
</appSettings>  
.....  
.....  
</configuration>  
\
```

Open the Report Viewer page in **Index.cshtml** page.

Replace the component tag **Html.EJ().ReportViewer("viewer")** with **Html.Bold().ReportViewer("viewer")**.

```
'js  
<div style="min-height:600px">  
@(Html.Bold().ReportViewer("viewer"))  
</div>  
\
```

Server-side migration

In the Solution Explorer, right-click the **References** and remove the **Syncfusion.EJ.ReportViewer** assembly reference.

Add the assembly from the Bold Reports NuGet package **BoldReports.Web**. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for **BoldReports.Web** NuGet package, and then install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Web API Controller

The `IReportController` interface is moved to `BoldReports.Web.ReportViewer`. Open the Report Viewer Web API Controller file and remove the following using statement.

```
'csharp
using Syncfusion.EJ.ReportViewer;
'
```

Add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Your application is successfully upgraded to the latest version of Report Viewer, and you can run the application with new assemblies.

Report export configuration

Now, the `BoldReports.Web` can export the reports with data visualization components only using web components. It is mandatory to configure the web scripts in Report Viewer Web API controller for exporting data visualization components such as chart, gauge, and map that are used in report definition. To configure the scripts in Web API, refer to the following steps:

Open the Report Viewer Web API controller.

Configure the following scripts and styles in `OnInitReportOptions` on Web API controller:

`jquery-1.10.2.min.js`

`bold.reports.common.min.js`

`bold.reports.widgets.min.js`

`ej.chart.min.js` - Exports the chart item. Add this script only if your report contains the chart report item.

`ej2-base.min.js`, `ej2-data.min.js`, `ej2-pdf-export.min.js`, `ej2-svg-base.min.js`, `ej2-lineargauge.min.js` and `ej2-circulargauge.min.js` - Exports the gauge item. Add this script only if your report contains the gauge report item.

`ej.map.min.js` - Exports the map item. Add this script only if your report contains the map report item.

bold.report-viewer.min.js

Replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        //Gauge component scripts
        resourcesPath + @"\bold-reports\common\ej2-base.min.js",
        resourcesPath + @"\bold-reports\common\ej2-data.min.js",
        resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
        resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
        resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
        resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
        //Map component script
        resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
        //Report Viewer Script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
    };
    reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
    {
        resourcesPath + @"\jquery-1.7.1.min.js"
    };
}
```

The data visualization components will not export without the above script configurations.

NuGet Packages for ASP.NET MVC

Refer to the following steps to configure Bold Reporting NuGet packages for ASP.NET MVC application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: [NuGet.org](#)

Source: <https://api.nuget.org/v3/index.json>

![Online NuGet Configure] (/static/assets/aspnet-mvc/report-viewer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reporting NuGet packages are shipped into our Bold Reporting Tools build. To configure the packages from Bold Reports installed location, use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages

![Offline NuGet Configure] (/static/assets/aspnet-mvc/report-viewer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

Installing NuGet packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**. Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages (**BoldReports.Mvc5** or **BoldReports.Web**), and then click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

install specified package in default project

```
Install-Package <Package Name>
```

install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

For example:

```
'cmd
```

install specified package in default project

```
Install-Package BoldReports.Web
```

install specified package in default project with specified Package Source

```
Install-Package BoldReports.Web –Source “https://api.nuget.org/v3/index.json”
```

install specified package in specified project

```
Install-Package BoldReports.Web -ProjectName BoldReportsApplication
```

Upgrading NuGet packages

Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**.
Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

Upgrading using Package Manager Console

To update the installed Bold Reporting NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

Update specific NuGet package in default project

```
Update-Package <Package Name>
```

Update all the packages in default project

```
Update-Package
```

Update specified package in default project with specified package source

```
Update-Package <Package Name> -Source <Source Location>
```

Update specified package in specified project

```
Update-Package <Package Name> - ProjectName <Project Name>
```

For example:

```
'cmd
```

Update specified Bold Reporting NuGet package

```
Update-Package BoldReports.Web
```

Update specified package in default project with specified Package Source

```
Update-Package BoldReports.Web –Source "https://api.nuget.org/v3/index.json"
```

Update specified package in specified project

```
Update-Package BoldReports.Web -ProjectName BoldReportsApplication
```

Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing `nuget.exe` to latest version use the following command:

```
'cmd  
nuget update -self  
'
```

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting NuGet packages.

```
'cmd
```

update all NuGet packages from config file

`nuget update <configPath> [options]`

update all NuGet packages from specified Packages Source

`nuget update -Source <Source Location> [optional]`

`configPath` is optional. It identifies the `package.config` or solutions file lists the packages utilized in the project.

For example:

```
'cmd
```

Update all NuGet packages from config file

`nuget update "C:\Users\BoldReportsApplication\package.config"`

Update all NuGet packages from specified Packages Source

`nuget update -Source "https://api.nuget.org/v3/index.json"`

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports ASP.NET MVC Report Viewer.

[How can improve the performance and handle the large amounts of data with Report Viewer?](#)

[Is Report Viewer compatible with latest version of JQuery library?](#)

[Is the back action from drillthrough report will load the report again with Report Viewer?](#)

[Is possible to change the culture of the date time parameter?](#)

[Is it possible to hide report parameters?](#)

[Is it possible to hide the export options in Report Viewer?](#)

[Is it possible to load reports providing parameter values at runtime?](#)

Is possible to change the culture of the date time parameter

Yes, we can change the culture of the date time parameter for Report Viewer by changing the locale. You can make use the following reference to change the locale of Report Viewer.

[ReportViewer Localization](#)

In Report Viewer, we could not change the culture of specific parameter or UI. So, we have to achieve the requirements only by changing the culture.

Is it possible to hide the parameters in Report Viewer

Yes, it is possible to hide the parameters in Report Viewer. On hiding the parameters, the users will not be able to have the parameter block in the Report Viewer. Refer to this [Hide parameter block and toolbar items](#) section.

Is it possible to hide the export options in Report Viewer

Yes, it is possible to hide the export options in Report Viewer. The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. Refer to this [Decide or Hide the export options](#) section.

Is it possible to load reports providing parameter values at runtime

Yes, it is possible to load reports with application inputs as parameters in Report Viewer. You need to provide the input values to the Report Viewer from client side at runtime using the `parameters` property, which accepts JSON array values. Refer to this [Set parameter at client](#) section.

How to Create RDL/RDLC Report

The following sections explain about how to create a new RDL/RDLC report using Bold Report Designer, Microsoft Report Builder and Visual Studio Report Server project template.

[Create a RDL report](#)

[Create a RDLC report](#)

[Change the exporting document file name based on parameter](#)

[Change the connection string datasource dynamically](#)

[Disable the vertical scrollbar in parameter panel](#)

Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

Bold Reports Report Designer.

Microsoft Report Builder.

Visual Studio Report Server project template.

Bold Reports Report Designer

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

Microsoft SQL Report Builder

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

Visual Studio Report Server template

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

Create a RDLC report using business object data source

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

Prerequisites

Microsoft Visual Studio 2017 or higher

[Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

Create business object class

Open Visual Studio from the File menu and select **New Project**.

Create project with class library type from the project type list.

![Add a new class library project](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/class-library-project.png)

Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
    public string ProdCat { get; set; }
    public string SubCat { get; set; }
    public string OrderYear { get; set; }
    public string OrderQtr { get; set; }
    public double Sales { get; set; }
}
```

Clean and build the application.

[Add an RDLC report](#)

Right-click the project and click **Add > New Item**.

Search Report with new item and select **Report Wizard** to start the report creation with dataset selection.

Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

[Data source and table configuration wizard](#)

Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/choose-data-source-type.png)

Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/select-data-objects.png)

In the DataSet Properties wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/arrange-table-fields.png)

Choose the table layout and click **Next**.

Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/aspnet-mvc/report-viewer/images/how-to/create-report/sales-report-design.png)

Adding data visualization scripts

To render the report with data visualization components such as chart, gauge and map items, must add scripts of the visualization element. The following table shows the script reference that need to be added in Report Viewer page for data visualization elements.

Visualization Item | Script File

Chart | ej.chart.min.js

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item, add chart control script **ej.chart.min.js** before the **bold.report-viewer.min.js** reference in **\Views\Shared_Layout.cshtml** page as in following code sample.

```
'html
<link href="~/Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="~/Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="~/Scripts/bold-reports/bold.report-viewer.min.js"></script>
```

The following code can be used to render the chart, gauge and map report items in Report Viewer.

```
'html
<link href="~/Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="~/Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="~/Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script, only if your report contains the gauge report item. -->
<script src="~/Scripts/bold-reports/common/ej2-base.min.js"></script>
<script src="~/Scripts/bold-reports/common/ej2-data.min.js"></script>
<script src="~/Scripts/bold-reports/common/ej2-pdf-export.min.js"></script>
<script src="~/Scripts/bold-reports/common/ej2-svg-base.min.js"></script>
<script src="~/Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js"></script>
<script src="~/Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js"></script>
<!--Used to render the map item. Add this script, only if your report contains the map report item.-->
<script src="~/Scripts/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="~/Scripts/bold-reports/bold.report-viewer.min.js"></script>
'
```

How to change the exporting document file name based on parameter

Find the following steps to change the file based on parameter values in Report.

Create the file exporting file name using the parameters in **onRenderingComplete** event and store it in local variable.

```
'html
function onRenderingComplete(event) {
    var parameters = event.reportParameters;
    if(parameters){
        for (var i = 0; i < parameters.length; i++) {
            if(parameters[i].Name == "Department"){
                this.exportFileName = "Sales for " + parameters[i].Value;
            }
        }
    }
}'
```

```
}
```

```
}
```

```
'
```

Use the file Name property with export, click event to change the file using the value stored in local variable used for having the file using the parameters.

```
'html
```

```
function onExportItemClick(event) {
```

```
event.fileName = this.exportFileName ;
```

```
}
```

```
'
```

How to change the connection string datasource dynamically

Find the following steps to change the connection string in datasource.

You need a report action information to get the data source information from report. So, stored the JsonResult information to local property in **PostReportAction** method as shown in the following code example.

```
'csharp
```

```
private Dictionary<string, object> _jsonResult;
```

```
//Post action for processing the rdl/rdlc report
```

```
public object PostReportAction(Dictionary < string, object > jsonResult)
```

```
{
```

```
if (jsonResult != null && jsonResult.Keys.Count > 0)
```

```
{
```

```
_jsonResult = jsonResult;
```

```
}
```

```
return ReportHelper.ProcessReport(jsonResult, this);
```

```
}
```

```
'
```

Use the JsonResult information with **ReportHelper.GetDatasource** API to get the data source details of the reports and you have to use the **DataSourceCredentials** to change the connection string of the data source.

```
'csharp
```

```
public void OnReportLoaded(ReportViewerOptions reportOption)
```

```
{  
if (jsonResult != null && jsonResult.Keys.Count > 0)  
{  
List<DataSourceInfo> datasources = ReportHelper.GetDataSources(_jsonResult, this);  
foreach (DataSourceInfo item in datasources)  
{  
if (item.DataProvider == "SQL")  
{  
string connectionString = "Data Source = dataplatformdemodata.syncfusion.com; Initial Catalog = AdventureWorks; User ID = 'demoreadonly@data-platform-demo'; Password = 'N@c)=Y8s*1&dh"';  
DataSourceCredentials DataSourceCredentials = new DataSourceCredentials();  
DataSourceCredentials.Name = item.DataSourceName;  
DataSourceCredentials.UserId = null;  
DataSourceCredentials.Password = null;  
DataSourceCredentials.ConnectionString = connectionString;  
DataSourceCredentials.IntegratedSecurity = false;//if windows credentials means we need to pass true  
as IntegratedSecurity  
reportOption.ReportModel.DataSourceCredentials = new List<DataSourceCredentials>  
{  
DataSourceCredentials  
};  
}  
}  
}  
}  
}  
`
```

How to disable the vertical scrollbar in parameter panel

To disable the vertical scrollbar in parameter panel, set the `enableparameterblockscroller` property to false.

```
<span style="font-weight:bold">Example</span>  
'js  
<div id="report viewer"></div>  
<script>  
$("#report viewer").boldReportViewer(
```

```
{  
enableParameterBlockScroller: false  
});  
</script>  
\
```

Bold Report Writer

Report Writer is a class library that enables the user to render reports defined in Microsoft's RDL format (2008 or 2008 R2) as PDF, Word, HTML, Excel or CSV documents.

The important features of MVC Report Writer are listed as follows:

RDL Specification - Supports RDL specification for the SQL Server 2008 and RDL specification for the SQL Server 2008 R2 only. List of available report definition formats:
[msdn.microsoft.com/library/dd297486\(SQL.100\).](https://msdn.microsoft.com/library/dd297486(SQL.100).)

Data sources - You can use advanced database servers Data Sources in Report Writer (SQL and Oracle).

Charts - Show all basic types of charts that are available in Microsoft RDL reports.

Tablix - Shows the summaries and simple tables.

Gauge - Shows measurement values using the expression values.

Textbox - Shows textbox data with expression support.

Export - Export report as PDF, Word, Excel, HTML, and CSV.

Report Parameter - Views the report based on the report parameter value.

Export SSRS RDL Report in Bold Reports ASP.NET MVC Report Writer

The Report Writer is a class library that is used to export the RDL report with popular file formats like PDF, Microsoft Word, Microsoft CSV, and Microsoft Excel without previewing the report in webpage. This section describes how to export the RDL report in ASP.NET MVC application using the [Report Writer](#).

Create an ASP.NET MVC application

Start the Visual Studio 2019 and click **Create new project**.

Choose **ASP.NET Web Application (.NET Framework)**, and then click **Next**.

Change the project name, and then click **Create**.

Choose the MVC and Web API, and then click OK.

![Creating a new ASP.NET MVC Application Project](/static/assets/aspnet-mvc/report-writer/images/getting-started/aspnet-mvc-web-application-template.png)

List of dependency libraries

In the Solution Explorer tab right click the project or solution , and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **BoldReports.Web** package, and install this in your MVC application.

The following table provides details about the dependency packages and their usage.

Package | Purpose

Syncfusion.Pdf.AspNet | Exports the report to a PDF.

Syncfusion.DocIO.AspNet | Exports the report to a Word.

Syncfusion.XlsIO.AspNet | Exports the report to an Excel.

Syncfusion.Presentation.AspNet | Exports the report to a PowerPoint.

Newtonsoft.Json | Serializes and deserialize data for the Report Writer. It is a mandatory package for Report Writer, and the package version should be 10.0.1 or higher.

In this tutorial, the **sales-order-detail.rdl** report is used and it can be downloaded [here](#). You can get the reports from Bold Reports installation location. For more information, refer to [samples and demos](#) section.

Server side Report Writer changes

Create a folder **Resources** in your application. Copy and paste the sample RDL reports into the **Resources** folder.

Open the **HomeController.cs** and add the following using statement.

```
'csharp
using System.IO;
using BoldReports.Writer;
'
```

Add the **Export()**function to load the report as stream. Refer to the following code snippet.

```
'csharp
[HttpPost]
public ActionResult Export(string writerFormat)
```

```
{
// Here, we have loaded the sales-order-detail sample report from application the folder
wwwroot\Resources.

FileStream reportStream = new
FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"\"Resources\sales-order-detail.rdl"),
 FileMode.Open, FileAccess.Read);

.....
}
```

Initialize the Report Writer instance with report stream and set the specified export format and file name to the export document.

```
`csharp
public ActionResult Export(string writerFormat)
{
.....
BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);
string fileName = null;
WriterFormat format;
string type = null;
fileName = "sales-order-detail.pdf";
type = "pdf";
format = WriterFormat.PDF;
}
`
```

You can use the **Save** method in Report Writer to generate the export document along with information of the report stream, it will return the generated file as Stream.

```
`csharp
MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);
// Download the generated export document to the client side.
memoryStream.Position = 0;
FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
```

```
return fileStreamResult;
```

```
'
```

Refer to the following complete code snippet used to get the exported file stream.

```
`csharp
[HttpPost]
public ActionResult Export(string writerFormat)
{
    // Here, we have loaded the sales-order-detail sample report from application the folder Resources.
    FileStream reportStream = new
    FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"\Resources\sales-order-detail.rdl"),
    FileMode.Open, FileAccess.Read);

    BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter(reportStream);

    string fileName = null;
    WriterFormat format;
    string type = null;

    if (writerFormat == "PDF")
    {
        fileName = "sales-order-detail.pdf";
        type = "pdf";
        format = WriterFormat.PDF;
    }
    else if (writerFormat == "Word")
    {
        fileName = "sales-order-detail.doc";
        type = "doc";
        format = WriterFormat.Word;
    }
    else if (writerFormat == "CSV")
    {
        fileName = "sales-order-detail.csv";
        type = "csv";
        format = WriterFormat.CSV;
    }
}
```

```
else if (writerFormat == "HTML")
{
    fileName = "sales-order-detail.html";
    type = "html";
    format = WriterFormat.HTML;
}

else if (writerFormat == "PPT")
{
    fileName = "sales-order-detail.ppt";
    type = "ppt";
    format = WriterFormat.PPT;
}

else
{
    fileName = "sales-order-detail.xls";
    type = "xls";
    format = WriterFormat.Excel;
}

MemoryStream memoryStream = new MemoryStream();
writer.Save(memoryStream, format);

// Download the generated export document to the client side.
memoryStream.Position = 0;

FileStreamResult fileStreamResult = new FileStreamResult(memoryStream, "application/" + type);
fileStreamResult.FileDownloadName = fileName;
return fileStreamResult;
}

`
```

Client side changes

Use the following code snippet in `Index.cshtml` home page to invoke the Web API from client side.

```
`html
@{Html.BeginForm("Export", "Home", FormMethod.Post);
{
```

```
<div>
<input type="submit" value="Generate" style="width: 150px;" />
</div>
}
}

Html.EndForm();
}
`
```

You can add the export file types to the home page to choose the format you want to export in the Report Writer. Copy and paste the following code snippet in your application.

```
`html
@{Html.BeginForm("Export", "Home", FormMethod.Post);
{
<div class="Common">
<div class="tablediv">
<div id="description_Pane" style="text-align: justify;">
<h3>Description</h3>
<span>
Bold ReportWriter is a powerful control for exporting RDL and RDLC files into specified format files.
</span>
</div>
<div class="rowdiv">
<div class="celldiv" style="padding:10px">
<label>
<strong> Save As :</strong>
</label>
<input id="rbtnPDF" type="radio" name="writerFormat" value="PDF" checked="checked" style="margin-left: 15px" />
<label for="rbtnPDF" style="padding:0px 5px 0px 2px">
PDF
</label>
<input id="rbtnWord" type="radio" name="writerFormat" value="Word" style="margin-left: 15px" />
<label for="rbtnWord" style="padding:0px 5px 0px 2px">
```

```

Word
</label>
<input id="rbtnxls" type="radio" name="writerFormat" value="xls" style="margin-left: 15px" />
<label for="rbtnxls" style="padding:0px 5px 0px 2px">
Excel
</label>
<input id="rbtnCSV" type="radio" name="writerFormat" value="CSV" style="margin-left: 15px" />
<label for="rbtnCSV" style="padding:0px 25px 0px 2px ">
CSV
</label>
<input class="buttonStyle" type="submit" name="button" value="Generate" style="width:150px;" />
</div>
</div>
</div>
</div>
</div>
Html.EndForm();
}
`
```

Now, Run and export the report with specified export format in your Report Writer application.

Congratulations! you have completed your first MVC Writer application!.Click [here](#) to download the already created MVC Report Writer application.

Reporting tools for ASP.NET MVC

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

Key features

Data Sources --- Supports connection to major data providers such as

Microsoft SQL Server, Oracle, OLEDB and **ODBC** for exploring data and design reports with a wide range of data sources.

User friendly Environment --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

Report items --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

Report Parameter --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

Expression --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

Add Web Report Designer to an ASP.NET MVC application

This section explains the steps required to add Web Report Designer to an ASP.NET MVC application.

Prerequisites

To get started with ASP.NET MVC 5 application, ensure the following software to be installed on the machine.

.Net Framework 4.5 and above.

ASP.NET MVC 5

Create ASP.NET MVC 5 web application

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![Project Creation Wizard](/static/assets/aspnet-mvc/report-designer/images/Getting-Started_img1.png)

Then choose the **MVC** in the template and enable the **Web API** option in the **Add folders and core references for:** section.

![Asp.Net MVC5 web application template](/static/assets/aspnet-mvc/report-designer/images/Getting-Started_img3.png)

Add Assembly References

Right-click the project/solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages....** Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Designer NuGet packages.

Search for **BoldReports.Web**, **BoldReports.Mvc5**, and **BoldReports.JavaScript** NuGet packages, and install them in your MVC application.

| Package | Purpose | |
|------------------------|-------------------------------------------------------------------------|--|
| BoldReports.Web | Used to create Web API service for processing the reports. | |
| BoldReports.Mvc5 | Contains tag helpers to create client-side web Report Designer control. | |
| BoldReports.JavaScript | Contains Report Designer scripts and style sheets. | |

Registering namespaces within Web.config

Open ~/Views/Web.config file and add the BoldReports.Mvc namespace under the namespaces tag.

```
'html
<namespaces>
<add namespace="BoldReports.Mvc"/>
</namespaces>
'
```

Disable unobtrusive mode

Set the **UnobtrusiveJavaScriptEnabled** to **false** in Web.config file of root directory as shown in the below image.

![Disable unobtrusive mode](/static/assets/aspnet-mvc/report-designer/images/Getting-Started_img17.png)

If you want to use with 'UnobtrusiveJavaScriptEnabled', then use **ej.unobtrusive.min.js** script with your application. You can get the script from the installed location as shown in the following image.

![unobtrusive script](/static/assets/aspnet-mvc/report-viewer/images/getting-started/unobtrusive-script.png)

Refer Scripts and Styles

Add the listed references in the same order as given below. You can replace the following code in the \Views\Shared_Layout.cshtml page.

```
'js
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>@ViewBag.Title - My ASP.NET Application</title>
@Styles.Render("~/Content/css")
@Styles.Render("~/Content/bold-reports/material/bold.reports.all.min.css")
@Styles.Render("~/Content/bold-reports/material/bold.reportdesigner.min.css")
@Styles.Render("~/Scripts/CodeMirror/lib/codemirror.css")
```

```
@Styles.Render("~/Scripts/CodeMirror/addon/hint/show-hint.css")
@Scripts.Render("~/bundles/modernizr")
<!--Render the gauge item. Add these scripts only if your report contains the gauge report item.-->
@Scripts.Render("~/Scripts/bold-reports/common/ej2-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-data.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-pdf-export.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/ej2-svg-base.min.js")
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js")
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js")
</head>
<body>
<div style="height: 600px; width: 100%;">
@RenderBody()
</div>
@Scripts.Render("~/bundles/jquery")
@Scripts.Render("~/bundles/bootstrap")
@Scripts.Render("~/Scripts/CodeMirror/lib/codemirror.js")
@Scripts.Render("~/Scripts/CodeMirror/addon/hint/show-hint.js")
@Scripts.Render("~/Scripts/CodeMirror/addon/hint/sql-hint.js")
@Scripts.Render("~/Scripts/CodeMirror/mode/sql/sql.js")
@Scripts.Render("~/Scripts/CodeMirror/mode/vb/vb.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.common.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.widgets.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.report-designer-widgets.min.js")
<!--Renders the chart item. Add this script, only if your report contains the chart report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.chart.min.js")
<!-- Report Designer component script-->
@Scripts.Render("~/Scripts/bold-reports/bold.report-viewer.min.js")
@Scripts.Render("~/Scripts/bold-reports/bold.report-designer.min.js")
@RenderSection("scripts", required: false)
</body>
</html>
`
```

Refer to the [Dependencies](#) to learn more details about web Report Designer dependent scripts and style sheets links.

Configure Script Manager

Open the `~/Views/Shared/_Layout.cshtml` page and add the Script Manager at the end of `<body>` element as in the following code sample.

```
'html
<body>
....
....
<!-- Bold Reporting ScriptManager -->
@Html.Bold().ScriptManager()
</body>
'
```

The main reason for referring the Script manager in `_Layout file` is that, it can be referred as common by all the View files present within your application.

Add Control in View page

Using `Bold()` tag add the Bold Report Designer component in any web page (`cshtml`) of your application in the `~/Views` folder.

```
'js
@(Html.Bold().ReportDesigner("designer"))
'
```

Add API controller

The MVC ReportDesigner uses WebApi services to process the report file and process the request from control.

Right-Click on the project and select `Add` then click `New Folder` and provide the folder name.

Right-Click on the newly created folder and select `Add` then click `New Item`.

Select `Web API Controller Class` from the listed templates and name the controller as `ReportingAPIController.cs`.

`![Provide controller name](/static/assets/aspnet-mvc/report-designer/images/Getting-Started_img19.png)`

Click Add.

`![Click Add](/static/assets/aspnet-mvc/report-designer/images/Getting-Started_img20.png)`

While adding Web API Controller class, name it with the suffix `Controller` that is mandatory.

Configure Web API

The interface **IReportDesignerController** has the declaration of action methods that are defined in Web API Controller to retrieve data, save, edit and browse reports from the server. The **IReportDesignerController** has the following action methods declaration.

| Methods | Description |
|---------------------|------------------------------------------------------------------------------------|
| PostDesignerAction | Action (HttpPost) method for posting the request for designer actions. |
| UploadReportAction | Action (HttpPost) method for posted file actions. |
| SetData | Write the resource into storage location. |
| GetData | Read the resource from storage location. |
| GetImage | Action (HttpGet) method for getting resource for report images. |
| PostReportAction | Action (HttpPost) method for posting the request for report process. |
| GetResource | Action (HttpGet) method for getting resource for report. |
| OnInitReportOptions | Report initialization method that occurs when the report is about to be processed. |
| OnReportLoaded | Report loaded method that occurs when the report and sub report start loading. |

ReportDesignerHelper

The class **ReportDesignerHelper** contains helper methods that help to process Post or Get request from the web Report Designer control and returns the response to the web Report Designer control. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

ReportHelper

The class **ReportHelper** contains helper methods that help process Post or Get request for report preview action and returns the response to the web Report Designer. It has the following methods.

| Methods | Description |
|---------------|------------------------------------------------------|
| GetResource | Returns the report resource for the requested key. |
| ProcessReport | Processes the report request and returns the result. |

Open **ReportingAPIController.cs** file and replace the following code example.

```
'csharp
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using BoldReports.Web;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
using System.IO;
using System.Reflection;
using System.Web;
namespace ReportDesignerSample.Api
{
    public class ReportingAPIController : ApiController, IReportDesignerController
    {
        private string GetFilePath(string itemName, string key)
        {
            string targetFolder = HttpContext.Current.Server.MapPath("~/");
            targetFolder += "Cache";
            if (!Directory.Exists(targetFolder))
            {
                Directory.CreateDirectory(targetFolder);
            }
            if (!Directory.Exists(targetFolder + "\\\" + key))
            {
                Directory.CreateDirectory(targetFolder + "\\\" + key);
            }
            return targetFolder + "\\\" + key + "\\\" + itemName;
        }
        [System.Web.Http.ActionName("GetImage")]
        [AcceptVerbs("GET")]
        public object GetImage(string key, string image)
        {
            return ReportDesignerHelper.GetImage(key, image, this);
        }
    }
}
```

```
}

/// <summary>
/// Action (HttpPost) method for posting the request for designer actions.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing designer request.</param>
/// <returns>The object data.</returns>
public object PostDesignerAction(Dictionary<string, object> jsonData)
{
    //Processes the designer request and returns the result.
    return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);
}

public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
    errorMessage = string.Empty;
    if (itemData.Data != null)
    {
        File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
    }
    else if (itemData.PostedFile != null)
    {
        var fileName = itemId;
        if (string.IsNullOrEmpty(itemId))
        {
            fileName = Path.GetFileName(itemData.PostedFile.FileName);
        }
        itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));
    }
    return true;
}

public ResourceInfo GetData(string key, string itemId)
{
    var resource = new ResourceInfo();
    resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
```

```
return resource;
}

/// <summary>
/// Action (HttpPost) method for uploaded file actions.
/// </summary>
public void UploadReportAction()
{
    //Processes the designer file upload request's.

    ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}

/// <summary>
/// Action (HttpGet) method for getting resource to report.
/// </summary>
/// <param name="key">The unique key to get the required resource.</param>
/// <param name="resourcetype">The type of the requested resource.</param>
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]
public object GetResource(string key, string resourcetype, bool isPrint)
{
    //Returns the report resource for the requested key.

    return ReportHelper.GetResource(key, resourcetype, isPrint);
}

/// <summary>
/// Report Initialization method that is triggered when report begin processed.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
```

```
{  
resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",  
resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",  
//Chart component script  
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",  
//Gauge component scripts  
resourcesPath + @"\bold-reports\common\ej2-base.min.js",  
resourcesPath + @"\bold-reports\common\ej2-data.min.js",  
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",  
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",  
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",  
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",  
//Map component script  
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",  
//Report viewer Script  
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",  
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"  
};  
reportOption.ReportModel.ExportResources.DependentScripts = new List<string>  
{  
resourcesPath + @"\jquery-1.7.1.min.js"  
};  
}  
/// <summary>  
/// Report loaded method that is triggered when report and sub report begins to be loaded.  
/// </summary>  
/// <param name="reportOptions">The ReportViewer options.</param>  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
//You can update report options here  
}  
/// <summary>  
/// Action (HttpPost) method for posting the request for report process.
```

```
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
    //Processes the report request and returns the result.
    return ReportHelper.ProcessReport(jsonData, this as IReportController);
}
```

Add routing information

Open the `WebApiConfig.cs` file from `App_Start` folder of your application.

Modify the `routeTemplate` in `Register` event to include the `{action}` parameter in the URI as follows.

```
'csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web.Http;
namespace ReportDesignerSample
{
    public static class WebApiConfig
    {
        public static void Register(HttpConfiguration config)
        {
            // Web API configuration and services
            // Web API routes
            config.MapHttpAttributeRoutes();
            config.Routes.MapHttpRoute(
                name: "DefaultApi",
                routeTemplate: "api/{controller}/{action}/{id}",
                defaults: new { id = RouteParameter.Optional } )
```

```
 );
}
}
}
`
```

Set the service URL

To browse, open and save the reports in the application, set the WebAPI controller name to the **ServiceUrl** property of the web Report Designer. You can replace the following code in the **cshtml** page.

```
'js
@(Html.Bold().ReportDesigner("designer").ServiceUrl(Url.Content("~/api/ReportingAPI")))
`
```

Run the Application

On running the application, web Report Designer will be rendered like below.

![Web Report Designer](/static/assets/aspnet-mvc/report-designer/images/Getting-Started_img7.png)

Dependencies

The ReportDesigner have the following list of internal and external dependencies to render the component.

The **BoldReports.JavaScript** contains reporting components scripts and style sheets. Install the **BoldReports.JavaScript** package, the scripts and styles required for Report Designer will be added to Scripts, Content folders in your application.

Scripts

Internal dependencies

| Name | Details | Local file path |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| bold.reports.common.min.js | Common script for reporting widgets. | ~/Scripts/bold-reports/common/bold.reports.common.min.js |
| bold.reports.widgets.min.js | Supports Syncfusion widgets to render in HTML5 format and it contains the dependent Syncfusion controls that is common for both report designer and viewer. | ~/Scripts/bold-reports/common/bold.reports.widgets.min.js |
| bold.report-designer-widgets.min.js | Supports Syncfusion widgets to render in HTML5 format and it contains the | ~/Scripts/bold-reports/common/bold.report-designer-widgets.min.js |

| | | |
|-----------------------------|---------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| | dependent Syncfusion controls for report designer | |
| ej.chart.min.js | Renders the chart item. Add this script, only if your report contains the chart report item. | ~/Scripts/bold-reports/data-visualization/ej.chart.min.js |
| ej2-base.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | ~/Scripts/bold-reports/common/ej2-base.min.js |
| ej2-data.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | ~/Scripts/bold-reports/common/ej2-data.min.js |
| ej2-pdf-export.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | ~/Scripts/bold-reports/common/ej2-pdf-export.min.js |
| ej2-svg-base.min.js | Renders the gauge item. Add this script only if your report contains the gauge report item. | ~/Scripts/bold-reports/common/ej2-svg-base.min.js |
| ej2-lineargauge.min.js | Renders the linear gauge item. Add this script only if your report contains the linear gauge report item. | ~/Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js |
| ej2-circulargauge.min.js | Renders the circular gauge item. Add this script only if your report contains the circular gauge report item. | ~/Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js |
| ej.map.min.js | Renders the map item. Add this script only if your report contains the map report item. | ~/Scripts/bold-reports/data-visualization/ej.map.min.js |
| bold.report-viewer.min.js | Used to preview the reports in report designer. | ~/Scripts/bold-reports/bold.report-viewer.min.js |
| bold.report-designer.min.js | Used to render the Bold Report Designer widget. | ~/Scripts/bold-reports/bold.report-designer.min.js |

External dependencies

| Name | Details | CDN link / Local file path |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| jQuery 1.7.1 and later versions | Common jQuery script to render the Bold Reporting widgets | Secured Link: https://cdn.boldreports.com/external/jquery-1.10.2.min.js Unsecured Link: http://cdn.boldreports.com/external/jquery-1.10.2.min.js |
| jsrender | The script to render the templates in the browser. | Secured Link: https://cdn.boldreports.com/external/jsrender.min.js Unsecured Link: http://cdn.boldreports.com/external/jsrender.min.js |
| Codemirror | Report Designer requires the code mirror scripts to edit the SQL queries and Visual Basic code functions with syntax highlighter. codemirror.js show-hint.js sql-hint.js sql.js vb.js | ~/Scripts/CodeMirror/lib/codemirror.js ~/Scripts/CodeMirror/addon/hint/show-hint.js ~/Scripts/CodeMirror/addon/hint/sql-hint.js ~/Scripts/CodeMirror/mode/sql/sql.js ~/Scripts/CodeMirror/mode/vb/vb.js |

Styles

Internal dependencies

| Name | Details | Local file path |
|-----------------------------|-------------------------------------------------------------------------|-----------------------------------------------------------------|
| bold.reports.all.min.css | It contains the styles and css references of dependent components. | ~/Content/bold-reports/[theme-name]/bold.reports.all.min.css |
| bold.reportdesigner.min.css | It contains the styles and css references of Report Designer component. | ~/Content/bold-reports/[theme-name]/bold.reportdesigner.min.css |

Refer the following syntax:

~/Content/bold-reports/[theme-name]/[fileName]

Example

| Theme Name | Format |
|---------------|-------------------------------------------------------------------------------------------------------------------------|
| Default-Theme | ~/Content/bold-reports/material/bold.reports.all.min.css ~/Content/bold-reports/material/bold.reportdesigner.min.css |

External dependencies

| Name | Details | Local file path |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| Codemirror | Report Designer requires the code mirror styles to edit the SQL queries and Visual Basic code functions with syntax highlighter. codemirror.css show-hint.css | ~/Scripts/CodeMirror/lib/codemirror.css ~/Scripts/CodeMirror/addon/hint/show-hint.css |

Service side dependencies

Internal dependencies

Package | Purpose

Syncfusion.DocIO.Base | Exports the report to a Word.

Syncfusion.Pdf.Base | Exports the report to a PDF.

Syncfusion.XlsIO.Base | Exports the report to an Excel.

Syncfusion.Presentation.Base | Exports the report to a PPT.

External dependencies

Package | Purpose

Newtonsoft.Json | Serializes and deserialize data for the Report Designer. It is a mandatory package for Report Designer, and the package version should be higher than 10.0.1 for NET Core 2.0 and others should be higher than 9.0.1.

Localization

Localization is the process of customizing the application for a given culture and locale - involving much more than the simple translation of text. In web report designer, localized strings appropriate to each culture are stored in separate files and loaded according to the UI culture setting.

By default report designer display strings in US English(en-US).

To render the static text with specific culture, refer the following corresponding culture script files and set culture name to the **locale** property of the Report Designer.

 ej.localetexts.fr-FR.min.js

```
<li> ej.culture.fr-FR.min.js </li>
</ul>
```

Install `BoldReports.Global` Nuget package in your MVC application.

Refer to the `ej.localetexts.fr-FR.min.js` script file from the `Scripts` folder of your application.

```
'html
<script src="~/Scripts/bold-reports/I10n/reportdesigner/ej.localetexts.fr-FR.min.js"></script>
`
```

Refer to the `ej.culture.fr-FR.min.js` script file from the `Scripts` folder of your application.

```
'html
<script src="~/Scripts/bold-reports/i18n/ej.culture.fr-FR.min.js"></script>
`
```

To render the localization Report Designer, use the following code snippet.

The following code example illustrates how to set the `locale` property for Report Designer in the `Index.cshtml` page.

```
'js
@using BoldReports.Mvc
@{
    ViewBag.Title = "Index";
}
<div style="height:600px">
    @Html.Bold().ReportDesigner("designer").ServiceUrl("/api/ReportingAPI").Locale("fr-FR"))
</div>
`
```

The following code example illustrates how to add Report Designer localization scripts in the layout page.

```
'html
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>@ViewBag.Title - My ASP.NET Application</title>
@Styles.Render("~/Content/css")
@Styles.Render("~/Content/bold-reports/material/bold.reports.all.min.css")
@Styles.Render("~/Content/bold-reports/material/bold.reportdesigner.min.css")
@Styles.Render("~/Scripts/CodeMirror/lib/codemirror.css")
@Styles.Render("~/Scripts/CodeMirror/addon/hint/show-hint.css")
@Scripts.Render("~/bundles/modernizr")
</head>
<body style="height:100%;width:100%;padding:0;">
@Scripts.Render("~/bundles/jquery")
@Scripts.Render("~/bundles/bootstrap")
@Scripts.Render("~/Scripts/CodeMirror/lib/codemirror.js")
@Scripts.Render("~/Scripts/CodeMirror/addon/hint/show-hint.js")
@Scripts.Render("~/Scripts/CodeMirror/addon/hint/sql-hint.js")
@Scripts.Render("~/Scripts/CodeMirror/mode/sql/sql.js")
@Scripts.Render("~/Scripts/CodeMirror/mode/vb/vb.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.common.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.reports.widgets.min.js")
@Scripts.Render("~/Scripts/bold-reports/common/bold.report-designer-widgets.min.js")
<!--Renders the chart item. Add this script, only if your report contains the chart report item.-->
@Scripts.Render("~/Scripts/bold-reports/data-visualization/ej.chart.min.js")
<!-- Report Designer component script-->
@Scripts.Render("~/Scripts/bold-reports/bold.report-viewer.min.js")
@Scripts.Render("~/Scripts/bold-reports/bold.report-designer.min.js")
@Scripts.Render("~/Scripts/bold-reports/I10n/reportdesigner/ej.localetexts.fr-FR.min.js")
@Scripts.Render("~/Scripts/bold-reports/i18n/ej.culture.fr-FR.min.js")
<div class="container body-content" style="height:100%;width:100%;">
@RenderBody()
@Html.Bold().ScriptManager()
</div>
@RenderSection("scripts", required: false)
```

```
</body>
</html>
`
```

Integrate the component with Report Server

You can integrate Report Designer with Report Server to create, edit, browse and publish reports using the Report Server built-in API service.

The Report Designer requires the `serviceAuthorizationToken` to perform the API actions with Bold Report Server. Create a token for the user by using the server RESTful API, you can use the following code in `HomeController.cs` file to generate the token.

```
`csharp
public class HomeController : Controller
{
    public ActionResult Index()
    {
        return View();
    }
    public ActionResult About()
    {
        ViewBag.Message = "Your application description page.";
        return View();
    }
    public ActionResult Contact()
    {
        ViewBag.Message = "Your contact page.";
        return View();
    }
    public string GenerateToken(string serverUrl, string userName, string password)
    {
        using (var client = new HttpClient())
        {
            client.DefaultRequestHeaders.Accept.Clear();
            var content = new FormUrlEncodedContent(new[]
            {
```

```

new KeyValuePair<string, string>("grant_type", "password"),
new KeyValuePair<string, string>("username", userName),
new KeyValuePair<string, string>("password", password)
});

var result = client.PostAsync("https://on-premise-
demo.boldreports.com/reporting/api/site/site1/token", content).Result;
string resultContent = result.Content.ReadAsStringAsync().Result;
var token = JsonConvert.DeserializeObject<Token>(resultContent);
return token.token_type + " " + token.access_token;
}
}
}

public class Token
{
    public string access_token { get; set; }
    public string token_type { get; set; }
    public string expires_in { get; set; }
    public string userName { get; set; }
    public string serverUrl { get; set; }
    public string password { get; set; }
}
`
```

Set the Bold Report Server built-in service URL to the `serviceUrl` property and assign the created token to `serviceAuthorizationToken` property. You can use the following complete code in your CSHTML page.

```

`js
@{
    var controller = ViewContext.Controller as ReportDesignerSample.Controllers.HomeController;
    var apiToken = controller.GenerateToken("https://on-premise-
        demo.boldreports.com/reporting/api/site/site1", "guest@boldreports.com", "demo");
}

<div style="height:500px">
    @(Html.Bold().ReportDesigner("designer"))
`
```

```
.ServiceAuthorizationToken(@apiToken)
.ServiceUrl(@"https://on-premise-demo.boldreports.com/reporting/reportservice/api/Designer")
.AjaxBeforeLoad("onAjaxRequest")
)
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'serverurl', Value: 'https://on-premise-
demo.boldreports.com/reporting/api/site/site1' });
}
</script>
</div>
`
```

Designing Reports

The Bold Report Designer provides an end-user documentation that describes how to interact with the Report Designer's UI and design an interactive reports. Refer the following user guide sections to get start with Bold Report Designer.

Connecting your data

A report must have a data source and dataset to include data. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

[Manage data source](#)

[Manage data set](#)

Transform your data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)

[Formatting columns](#)

[Query filter](#)

[Data set parameter](#)

[Query parameter](#)

[Configure expression columns](#)

Working with report parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it.

[Add report parameter](#)

[Edit report parameter](#)

[Delete report parameter](#)

[Cascading parameter](#)

[Multi-value parameter](#)

Embed images into the report

Supports to embed local or database images into the report and image data is stored within the report definition. The embedded images in a report are listed in the **Image Manager panel**.

[Embed images into the report](#)

Interacting with report design surface

WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

[Interacting with report design surface](#)

Report items

Offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

[Configure report items](#)

Customize appearance

The Properties panel shows all the properties of the selected section, report items or the report itself to customize the appearance of the report.

[Customize appearance](#)

Report design settings

[Configure design settings](#)

Shape report data

[Filter data](#)

[Group data](#)

[Sort data](#)

[Format data](#)

Interactive features

[Hyperlink](#)

[Drilldown](#)

[Interactive sorting](#)

Working with Expressions

Expressions are used to specify criteria for retrieving and formatting data, creating calculated fields and calculating summaries, conditionally shaping data and changing a report control's appearance.

[Expressions builder](#)

[Open report](#)

[Open report](#)

[Save report](#)[Save report](#)[Preview report](#)[Preview report](#)

Migrate Report Designer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Designer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Designer from Syncfusion Essential Studio release version to Bold Reports version of ASP.NET MVC Report Designer application.

Scripts

| Old Scripts | New Scripts | Description |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej.web.all.min.js | bold.reports.common.min.js bold.reports.widgets.min.js bold.report-designer-widgets.min.js ej.chart.min.js ej2-base.min.js ej2-data.min.js ej2-pdf-export.min.js ej2-svg-base.min.js ej2-lineargauge.min.js ej2-circulargauge.min.js ej.map.min.js bold.report-viewer.min.js | <p>We have removed the dependency scripts for report designer from existing ej.web.all.min.js script and provided the dependency scripts as below:</p> <ul style="list-style-type: none"> bold.reports.common.min.js - Common script for reporting widgets. bold.reports.widgets.min.js - It contains the scripts of dependent controls that are common for both Report Designer and Report Viewer. bold.report-designer-widgets.min.js - It contains the scripts of Report Designer dependent controls. ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart report item. ej2-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item. ej2-data.min.js - Renders the |

| | | |
|---------------------------------------|------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| | | <p>gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-pdf-export.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-svg-base.min.js</code> - Renders the gauge item. Add this script only if your report contains the gauge report item.</p> <p><code>ej2-lineargauge.min.js</code> - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.</p> <p><code>ej2-circulargauge.min.js</code> - Renders the circular gauge item. Add this script only if your report contains the circular gauge report item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Previews the reports designed with Report Designer.</p> |
| <code>ej.reportdesigner.min.js</code> | <code>bold.report-designer.min.js</code> | Renamed the report designer script file and it used to render the Bold Report Designer widget. |

Styles

| Old Scripts | New Scripts | Description |
|---------------------------------|---------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <code>ej.web.all.min.css</code> | <code>bold.reports.all.min.css</code> | We have removed the dependent controls styles for report designer from existing <code>ej.web.all.min.css</code> script and provided the |

| | | |
|--|--|--------------------------------------------------------|
| | | dependent controls styles as bold.reports.all.min.css. |
|--|--|--------------------------------------------------------|

Remove the old scripts and styles references from existing application, then add the new scripts and styles references based on above script name changes.

Assemblies

| Purpose | Old Assemblies | New Assemblies | Description |
|-----------------------|--------------------------------------------------------------------|---------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Base Assemblies | Syncfusion.EJ.ReportViewer.dll Syncfusion.EJ.ReportDesigner.dll | BoldReports.Web.dll | The Syncfusion.EJ.ReportViewer.dll and Syncfusion.EJ.ReportDesigner.dll assemblies have been combined as BoldReports.Web.dll. |
| MVC Helper Assemblies | Syncfusion.EJ.dll Syncfusion.EJ.MVC.dll | BoldReports.Mvc.dll | The Syncfusion.EJ.dll and Syncfusion.EJ.MVC.dll assemblies have been combined as BoldReports.Mvc.dll. |

Packages

| Purpose | Old Packages | New Packages |
|--------------------|----------------------------------------------------------------------------|--------------------------------------|
| Server Side Helper | Syncfusion.ReportDesigner.AspNet.Mvc Syncfusion.ReportViewer.AspNet.Mvc | BoldReports.Web |
| MVC Helper | Syncfusion.AspNet.Mvc4 Syncfusion.AspNet.Mvc5 | BoldReports.Mvc4 BoldReports.Mvc5 |

Register tag prefix in Web.config file

The ASP.NET MVC reporting components tag prefix has been changed from EJ() to Bold(). Open the ~/Views/Web.config file and add the BoldReports.Mvc assembly reference to the <system.web.webPages.razor> element.

| Old code snippet | New code snippet |
|-------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|
| <configuration> <system.web.webPages.razor> <pages> | <configuration> <system.web.webPages.razor> <pages> |

| | |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>pageBaseType="System.Web.Mvc.WebViewPage"> <namespaces> <add namespace="Syncfusion.EJ.MVC"/> </namespaces> </pages> </system.web.webPages.razor> </configuration></pre> | <pre>pageBaseType="System.Web.Mvc.WebViewPage"> <namespaces> <add namespace="BoldReports.Mvc"/> </namespaces> </pages> </system.web.webPages.razor> </configuration></pre> |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Namespace changes

| Assembly Name | Old Namespace | New Namespace |
|---------------------|------------------------------------------------------------------------|------------------------------------|
| BoldReports.Web.dll | Syncfusion.Reports.EJ | BoldReports.Web |
| | Syncfusion.EJ.ReportWriter | BoldReports.Writer |
| | Syncfusion.EJ.ReportViewer | BoldReports.Web.ReportViewer |
| | Syncfusion.EJ.ReportDesigner | BoldReports.Web.ReportDesigner |
| | Syncfusion.Reports.EJ.Data | BoldReports.Data |
| | Syncfusion.Reporting | BoldReports.Configuration |
| | Syncfusion.EJ.RDL.ServerProcessor | BoldReports.ServerProcessor |
| BoldReports.Mvc.dll | Syncfusion.JavaScript Syncfusion.MVC.EJSyncfusion.JavaScript.Shared | BoldReports.Web BoldReports.Mvc |

Based on above assembly and namespace changes, modify the Report Designer **Web API Controller** in your application.

Configure script manager

| Old code snippet | New code snippet |
|------------------------------|--------------------------------|
| @(Html.EJ().ScriptManager()) | @(Html.Bold().ScriptManager()) |

Control initialization

| Old code snippet | New code snippet |
|--------------------------------------|----------------------------------------|
| Html.EJ().ReportDesigner("designer") | Html.Bold().ReportDesigner("designer") |

Report export configuration

The **BoldReports.Web** assembly can export the reports with data visualization components such as chart, gauge, and map, only if we configure the web scripts in Report Designer Web API controller. To configure the scripts in Web API controller, refer to the following steps:

Open the Report Designer Web API controller.

Configure the following scripts and styles in **OnInitReportOptions** on Web API controller:

jquery-1.7.1.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js - Exports the chart item. Add this script only if your report contains the chart report item.

ej2-base.min.js, **ej2-data.min.js**, **ej2-pdf-export.min.js**, **ej2-svg-base.min.js**, **ej2-lineargauge.min.js** and **ej2-circulargauge.min.js** - Exports the gauge item. Add this script only if your report contains the gauge report item.

ej.map.min.js - Exports the map item. Add this script only if your report contains the map report item.

bold.report-viewer.min.js

You can replace the **OnInitReportOptions** action in Report Designer Web API controller using below code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
    reportOption.ReportModel.ExportResources.Scripts = new List<string>
    {
        resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
        resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
        //Chart component script
        resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
        //Gauge component scripts
        resourcesPath + @"\bold-reports\common\ej2-base.min.js",
        resourcesPath + @"\bold-reports\common\ej2-data.min.js",
        resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
        resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
    }
}
```

```
resourcesPath + @"\\bold-reports\\data-visualization\\ej2-lineargauge.min.js",
resourcesPath + @"\\bold-reports\\data-visualization\\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.map.min.js",
//Report Designer Script
resourcesPath + @"\\bold-reports\\data-visualization\\ej.chart.min.js",
resourcesPath + @"\\bold-reports\\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\\jquery-1.7.1.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

NuGet Packages for ASP.NET MVC

Refer to the following steps to configure Bold Reporting tools NuGet packages for ASP.NET MVC application.

Configure NuGet feed URL

Online NuGet feed URL

The Bold Reporting tools NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: `NuGet.org`

Source: `https://api.nuget.org/v3/index.json`

![Online NuGet Configure](/static/assets/aspnet-mvc/report-designer/images/nuget-packages/NuGet_Config.png)

Offline NuGet feed URL

Bold Reporting tools NuGet packages are shipped into our Report Platform SDK build. To configure the packages from Bold Reports installed location, use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

Name: Bold Reports installed NuGet

Source: {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages.

![Offline NuGet Configure](/static/assets/aspnet-mvc/report-designer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

Installing NuGet packages

Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**.

Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages (**BoldReports.Mvc5** or **BoldReports.Web**), and then click **Install** button.

Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
cmd
```

install specified package in default project

Install-Package <Package Name>

install specified package in default project with specified package source

Install-Package <Package Name> -Source <Source Location>

install specified package in specified project

Install-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

install specified package in default project

Install-Package BoldReports.Web

install specified package in default project with specified Package Source

Install-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

install specified package in specified project

Install-Package BoldReports.Web -ProjectName BoldReportingApplication

Upgrading NuGet packages

Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**

Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

Upgrading using Package Manager Console

To update the installed Bold Reporting tools NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

`cmd

Update specific NuGet package in default project

Update-Package <Package Name>

Update all the packages in default project

Upgrading NuGet packages

Update all the packages in default project

Update-Package

Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

For example:

`cmd

Update specified Bold Reporting NuGet package

Update-Package BoldReports.Web

Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportingApplication

Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing nuget.exe to latest version use the following command:

`cmd

nuget update -self

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting tools NuGet packages.

`cmd

update all NuGet packages from config file

Normal layout

update all NuGet packages from config file

nuget update <configPath> [options]

update all NuGet packages from specified Packages Source

nuget update -Source <Source Location> [optional]

\ configPath is optional. It identifies the package.config or solutions file lists the packages utilized in the project.

For example:

`cmd

Update all NuGet packages from config file

nuget update "C:\Users\SyncfusionApplication\package.config"

Update all NuGet packages from specified Packages Source

nuget update -Source "https://api.nuget.org/v3/index.json"

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

Responsive layout rendering of ASP.NET MVC Report Designer

Report Designer will adaptively render itself with optimal user interfaces for tablet or desktop form factors. It has built-in responsive support, so that it will adjust automatically based on the browser viewport. Setting width to 100% is simply enough to make it responsive. This helps your application to scale elegantly on all form factors with ease.

Normal layout

The following output shows the normal layout rendering of Report Designer tool bar items.

![Rendering of Report Designer tool bar items in normal layout](/static/assets/aspnet-mvc/report-designer/images/normal-layout.png)

Responsive layout

The following output shows the responsive layout rendering of Report Designer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/aspnet-mvc/report-designer/images/responsive-layout.png)

Samples and Demos

Browse and explore our ready-to-use the designer samples, online and offline demos.

Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

Online demos

You can view the ASP.NET MVC Report Designer online demo samples from [here](#).

GitHub demo samples

Click [here](#) to view the GitHub Report Designer demo samples.

Supported Browsers

This document provides the list of web browsers supported by the Web Report Designer

| Desktop Browsers | Version |
|-------------------|---------|
| Internet Explorer | 11.0+ |
| Microsoft Edge | Current |
| Firefox | 22+ |
| Google Chrome | 17+ |
| Opera | 12+ |
| Safari | 5+ |

How to queries for Bold Reports Report Designer

This section helps to get the answer for the frequently asked how to queries in Bold Reports Report Designer.

[Open server report using report path on opening Report Designer](#)

[How to add datasource and dataset for Report Designer from application?](#)

How to open the RDL file at the time of opening the Report Designer

Find the following steps to open server report on opening Report Designer.

Create a function and bind it with the `create` API in `Index.cshtml` file as like in below code snippet.

```
'html
<div style="height:500px">
@(Html.Bold().ReportDesigner("designer")
.Create("controlInitialized")
)
</div>
<script type="text/javascript">
function controlInitialized(args) {
...
}
```

```
}
```

```
</script>
```

```
,
```

Use the `openReport` method in the function along with report path that was previously created, as like in below code snippet.

```
'javascript
```

```
function controlInitialized(args) {
```

```
var designer = $('#designer').data('boldReportDesigner');
```

```
designer.openReport("/Sample Reports/Sales Report");
```

```
}
```

```
,
```

How to add the data source and dataset for Report Designer from application

You can add the data for reports in the Report Designer from application level on initializing the Report Designer. This can be achieved using the `addDataSource` and `addDataSet` functions, by which you can add the data source and dataset for the reports at the time of initialization of Report Designer.

Find the following steps to add the data source and dataset for Report Designer from application.

Create a function and bind it with the `create` API in `Index.cshtml` file as shown in the following code snippet.

```
'html
```

```
<div style="height:500px">
```

```
@(Html.Bold().ReportDesigner("designer"))
```

```
.Create("controlInitialized")
```

```
)
```

```
</div>
```

```
<script type="text/javascript">
```

```
function controlInitialized(args) {
```

```
...
```

```
}
```

```
</script>
```

```
,
```

Use the `addDatasource` method to add the data source in Report Designer as shown in the following code snippet,

```
`javascript
var datasource =
{
    type:'BoldReports.RDL.DOM.DataSource',
    Name:'DataSource1',
    Transaction:false,
    DataSourceReference:null,
    SecurityType:'DataBase',
    ConnectionProperties:{
        type:'BoldReports.RDL.DOM.ConnectionProperties',
        ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
        EmbedCredentials:false,
        DataProvider:'SQL',
        IsDesignState:false,
        IntegratedSecurity:false,
        UserName:'',
        PassWord:'',
        Prompt:'Specify the Username and password for DataSource DataSource1',
        CustomProperties: []
    }
};

function controlInitialized() {
    var designerObj = $("#designer").data("boldReportDesigner");
    designerObj.addDataSource(datasource);
}

`
```

Use the `addDataSet` method to add dataset in Report Designer as shown in the following code snippet,

```
`javascript
var dataset =
```

```
{  
    type:'BoldReports.RDL.DOM.DataSet',  
    Name:'DataSet1',  
    Fields:[  
        { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},  
        { type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},  
        { type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },  
        { type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null }  
    ],  
    Query: {  
        type: "BoldReports.RDL.DOM.Query",  
        CommandText: "SELECT  
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[Huma  
nResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM  
[HumanResources].[Department]",  
        CommandType: 0,  
        DataSourceName: "DataSource1",  
        QueryDesignerState: {  
            type: "BoldReports.RDL.DOM.QueryDesignerState",  
            Expressions: null,  
            Filters: null,  
            Joins: null,  
            StoredProcedure: null,  
            Tables: [  
                {  
                    type: "BoldReports.RDL.DOM.Table",  
                    Columns: [  
                        { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
                        IsSelected: true, Name: "DepartmentID"  
                    },  
                    { type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
                    IsSelected: true, Name: "Name"  
                }  
            ]  
        }  
    }  
}
```

```
    IsSelected: true, Name: "Name"  
},  
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
    IsSelected: true, Name: "GroupName"  
},  
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,  
    IsSelected: true, Name: "ModifiedDate"  
}  
],  
Name: "Department",  
Schema: "HumanResources",  
SchemaLevels: [  
    { Name: "HumanResources", SchemaType: "Schema"},  
    { Name: "Tables", SchemaType: "Category"},  
    { Name: "Department", SchemaType: "Table"}  
]  
}  
]  
}  
},  
QueryParameters: [],  
Timeout: 0  
},  
CaseSensitivity:0,  
Collation:null,  
AccentSensitivity:0,  
KanatypeSensitivity:0,  
WidthSensitivity:0,  
Filters:[],  
SharedDataSet:null,  
InterpretSubtotalsAsDetails:0,  
DataSetInfo:null,  
DataSetObject:null  
};
```

```
function controlInitialized() {
var designerObj = $("#designer").data("boldReportDesigner");
designerObj.addDataSet(dataset);
}
`
```

Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

Breaking Changes

When you upgrade from previous versions of Bold Reports to 2.2.23, there are few breaking changes. The following section explains the breaking changes for Bold Reports version 2.2.23.

Designer resource read and write API

The resource write and read API's in `IReportDesignerController` are modified to simplify the resource write and read actions with Bold Report Designer.

Below are the new API details tabulated against old API's.

| Old API | New API | Description |
|-------------|---------|-----------------------------------------------------|
| UploadFile | SetData | Writes the designer resource into storage location. |
| GetFilePath | GetData | Reads the designer resource from storage location. |
| GetFiles | | |
| GetFile | | |

Parameters

SetData

| Parameter Name | Type | Description |
|----------------|---------------|--------------------------------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |
| itemData | object | Gets the resource data. |
| | Property Name | |
| | Data | byte array Gets the resource data as byte array. To write |

| | | | | |
|--------------|------------|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|
| | | | an external resource into storage location, pass the resource data in this property. | |
| | PostedFile | HttpPostedFile | Gets the uploaded resource data as HttpPostedFile . The resource data uploaded within report designer will be received in this property. | |
| errorMessage | string | | | Returns the error message, if the write action is failed. |

GetData

| Parameter Name | Types | Description |
|----------------|--------|-------------------------------------------|
| key | string | Bold report designer unique token. |
| itemId | string | Unique id of designer resource. |

Return Type

| API Name | Return Type | |
|---------------|-------------|----------------------------------------------------------|
| SetData | boolean | |
| GetData | object | |
| Property Name | Type | Description |
| Data | byte array | Contains the requested resource data as byte array. |
| errorMessage | string | Returns the error message, if the read action is failed. |

Code snippet

| Old snippet | New snippet |
|----------------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| UploadFilecsharppublic bool UploadFile(System.Web.HttpPostedFile) | SetDatacsharppublic bool SetData(string key, string itemId, ItemInfo itemData, out string |

| | |
|---------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| <code>httpPostedFile){ //Implement resource write actions}</code> | <code>errorMessage){ //Implement resource write actions}</code> |
| <code>GetFiles csharppublic List GetFiles(FileType fileType){ //Implement resource read actions}</code> | |
| <code>GetFilePath csharppublic string GetFilePath(string fileName){ //Implement actions to get file path}</code> | <code>GetData csharppublic ResourceInfo GetData(string key, string itemId){ //Implement resource read actions}</code> |
| <code>GetFile csharppublic FileModel GetFile(string filename, bool isOverride){ //Implement resource read actions}</code> | |

Reporting tools for ASP.NET Web Forms

Enterprise-class reporting tools for ASP.NET Web Forms development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your web applications.

How to best read this user guide

The best way to get started would be to read the `Getting Started` section of the documentation for the control that you would like to start using first. The `Getting Started` guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

Reporting tools for ASP.NET Web Forms

Enterprise-class reporting tools for ASP.NET Web Forms development to embed reporting functionalities such as designing, viewing, exporting, and printing reports in your web applications.

How to best read this user guide

The best way to get started would be to read the `Getting Started` section of the documentation for the control that you would like to start using first. The `Getting Started` guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application. A good starting point would be to refer to the code snippets in the online [sample browser](#) which contains code samples, it is very likely that you will find a code sample that resembles your intended usage scenario.

Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

System Requirements

This topic describes the software and hardware requirements for setting up the development environment of Bold Reports ASP.NET Webforms.

Supported Operating Systems

Windows 7+, 8+

Windows Server 2008 R2+

Hardware Requirements

The following hardware requirements are necessary for setting up the development environment of Bold Reports ASP.NET Web Forms:

1 GHZ or faster, 32 bit or 64 bit processor.

1 GB RAM for 32 bit or 2 GB RAM for 64 bit.

Software Requirements

The following software requirements are necessary for setting up the development environment of Bold Reports ASP.NET Web Forms:

Microsoft Visual Studio 2010 or later

Internet Information Services (IIS) 7.0+

Framework

The below tool is required for Bold Reports ASP.NET Web Forms.

.NET Framework 4.5 or higher

Browser Compatibility

IE 9+

Microsoft Edge

Mozilla Firefox 22+

Chrome 17+

Opera 12+

Safari 5+

See Also

[Licensing procedure for deployment](#)

Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Microsoft Excel, CSV, Microsoft Word, and HTML formats. Some of the key features are,

Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.

Easily customize each element of Report Viewer and provide events for report processing customization.

Display ssrs rdl report in Bold Reports ASP.NET Web Forms Report Viewer

This section explains you the steps required to create your first ASP.NET Web Forms reporting application to display already created SSRS RDL report in Bold Reports ASP.NET Web Forms Report Viewer without using a Report Server.

Create ASP.NET Web Forms application

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![ASP.NET Web Forms application project template](/static/assets/aspnet-web-forms/report-viewer/images/getting-started/aspnetmvc5-template.png)

Choose **Web Forms, Web API** and then click **OK**.

![Select Web API and Web Forms options](/static/assets/aspnet-web-forms/report-viewer/images/getting-started/aspnet-config-template.png)

Configure Report Viewer in an application

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Display ssrs rdl report in Bold Reports ASP.NET Web Forms Report Viewer Configure Report Viewer in an application

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web** and **BoldReports.WebForms** NuGet packages, and install them in your Web Forms application. The following table provides details about the packages and their usage.

Package | Purpose

PostReportAction | Action (HttpPost) method for posting the request in report process.

OnInitReportOptions | Report initialization method that occurs when the report is about to be processed.

OnReportLoaded | Report loaded method that occurs when the report and sub report start loading.

GetResource | Action (HttpGet) method to get resource for the report.

[ReportHelper](#)

The class **ReportHelper** contains helper methods that help to process a Post or Get request from the Report Viewer control and return the response to the Report Viewer control. It has the following methods:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

[Add Web API Controller](#)

Right-click **Controller** folder in your project and select **Add > New Item** from the context menu.

Select **Web API Controller Class** from the listed templates and name it as
ReportViewerController.cs

![Provide controller name](/static/assets/aspnet-web-forms/report-viewer/images/getting-started/add-web-api-controller.png)

Click **Add**.

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Display ssrs rdl report in Bold Reports ASP.NET Web Forms Report Viewer Configure Report Viewer in an application

Inherit the **IReportController** interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
public class ReportViewerController : ApiController, IReportController
{
    // Post action for processing the RDL/RDLC report
    public object PostReportAction(Dictionary<string, object> jsonResult)
    {
        return ReportHelper.ProcessReport(jsonResult, this);
    }

    // Get action for getting resources from the report
    [System.Web.Http.ActionName("GetResource")]
    [AcceptVerbs("GET")]
    public object GetResource(string key, string resourcetype, bool isPrint)
    {
        return ReportHelper.GetResource(key, resourcetype, isPrint);
    }

    // Method that will be called when initialize the report options before start processing the report
    public void OnInitReportOptions(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }

    // Method that will be called when reported is loaded
    public void OnReportLoaded(ReportViewerOptions reportOption)
    {
        // You can update report options here
    }
}
```

Add routing information

To configure routing to include an action name in the URI, open the **WebApiConfig.cs** file and change the **routeTemplate** in the **Register** method as follows,

```
'csharp
```

Display ssrs rdl report in Bold Reports ASP.NET Web Forms Report Viewer Set report path and service URL

```
public static class WebApiConfig
{
    public static void Register(HttpConfiguration config)
    {
        // Web API configuration and services
        // Web API routes
        config.MapHttpAttributeRoutes();
        config.Routes.MapHttpRoute(
            name: "DefaultApi",
            routeTemplate: "api/{controller}/{action}/{id}",
            defaults: new { id = RouteParameter.Optional }
        );
    }
}
```

[Set report path and service URL](#)

To render the reports available in the application, set the `ReportPath` and `ReportServiceUrl` properties of the Report Viewer. You can replace the following code in your Report Viewer page.

```
`js
<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">
    <link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
    <script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
    <script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
    <script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
    <script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
    <div style="height: 650px; width: 950px; min-height: 404px;">
        <Bold:ReportViewer runat="server" ID="viewer" ReportPath="~/Resources/sales-order-detail.rdl"
            ReportServiceUrl="/api/ReportViewer">
        </Bold:ReportViewer>
    </div>
</asp:Content>
```

The report path property is set to the RDL report that is added to the project `Resources` folder.

Preview the report

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Sales Order Detail Report](/static/assets/aspnet-web-forms/report-viewer/images/getting-started/sales-order-detail.png)

See Also

[Render report with data visualization report items](#)

[Create RDLC report](#)

[List of SSRS server versions are supported in Bold Reports](#)

Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports, set the `reportServerUrl`, `reportPath`, and `reportServiceUrl` properties as shown in the following code snippet.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer">
  ReportServiceUrl="/api/SSRSReports"
  ReportPath="/SSRSSamples/Territory Sales"
  ReportServerUrl="http://<servername>/reportserver$instanceName">
</Bold:ReportViewer>
</div>
'
```

Report Server URL should be in the format of `http://<servername>/reportserver$instanceName`

The report path should be in the format of `/folder name/report name`.

Network credentials for SSRS

The network credentials are required to load specified SSRS report from the specified SSRS Report Server using the Report Viewer. Specify the `ReportServerCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server credential
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
        "RDLReport1");
}
```

Set data source credential to shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS Server. If the report has any data source that uses credentials to connect with the database, then you should specify the `DataSourceCredentials` for each report data source to establish database connection.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add SSRS Report Server and data source credentials
    reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
    "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));

}
```

Data source credentials should be added to the shared data sources that do not have credentials in the connection strings.

Render linked reports

You can render a linked report that points to an existing report, which is published in the SSRS Report Server. You can set the parameter, data source, credential, and other properties like normal SSRS reports using the Report Viewer.

'js

```
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/SSRSReports"
ReportPath="/SSRSSamples/Territory Sales_Link"
ReportServerUrl="http://<servername>/reportserver$instanceName">
</Bold:ReportViewer>
</div>
```

The `Territory Sales_Link` is a linked report created for the `Territory Sales` report that is already available in the SSRS Report Server.

Load SharePoint Server reports

To render SharePoint server reports, set the `reportServerUrl`, `reportPath`, and `reportServiceUrl` properties as shown in the following code snippet.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/SharePointReports"
ReportServerUrl="http://<servername>/reportserver$instanceName"
ReportPath="http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl">
</Bold:ReportViewer>
</div>
'
```

In SharePoint integrated mode, the `reportServerUrl` will be same as your site URL. The `reportPath` is relative to the Report Server URL with the file extension.

[Forms credential for SharePoint Server](#)

The forms credentials are required to load the SharePoint integrated SSRS report from the specified SharePoint integrated SSRS Report Server using the Report Viewer. Specify the `ReportServerFormsCredential` property in the Web API Controller `OnInitReportOptions` method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential for server
    reportOption.ReportModel.ReportServerFormsCredential = new
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");
}
```

[Set data source credential to shared data sources](#)

The shared data source credentials can be added to the `DataSourceCredentials` property to connect with the database.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    //Add ReportServerFormsCredential and data source credentials
    reportOption.ReportModel.ReportServerFormsCredential = new
    BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");

    reportOption.ReportModel.DataSourceCredentials.Add(new
    BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Load Bold Report Server reports

Set data source credential to shared data sources

Data source credentials should be added to shared data sources that do not have credentials in the connection strings.

Load Bold Report Server reports

You can render the Bold Report Server reports in the Report Viewer easily without creating a Web API service. Bold Report Server provides the built-in Web API service that helps you to display the server reports.

The Report Viewer requires the `ServiceAuthorizationToken` to connect and download the items from the Bold Report Server.

To load the Report Server reports, follow these steps:

Create a token for users by using the server RESTful API. The following code is used to set the `ServiceAuthorizationToken` value in your application `Page_Load()` function.

```
'csharp
public partial class _Default : Page
{
protected void Page_Load(object sender, EventArgs e)
{
this.viewer.ServiceAuthorizationToken = GenerateToken("guest@boldreports.com", "demo");
}
public string GenerateToken(string userName, string password)
{
using (var client = new HttpClient())
{
client.DefaultRequestHeaders.Accept.Clear();
var content = new FormUrlEncodedContent(new[]
{
new KeyValuePair<string, string>("grant_type", "password"),
new KeyValuePair<string, string>("username", userName),
new KeyValuePair<string, string>("password", password)
});
var result = client.PostAsync("https://on-premise-
demo.boldreports.com/reporting/api/site/site1/token", content).Result;
string resultContent = result.Content.ReadAsStringAsync().Result;
var token = JsonConvert.DeserializeObject<Token>(resultContent);
```

Load Bold Report Server reports

Set data source credential to shared data sources

```
return token.tokentype + " " + token.accesstoken;  
}  
}  
}  
}  
  
public class Token  
{  
  
    public string access_token { get; set; }  
    public string token_type { get; set; }  
    public string expires_in { get; set; }  
    public string userName { get; set; }  
    public string serverUrl { get; set; }  
    public string password { get; set; }  
}  
`
```

Set the Bold Report Server built-in service URL to the **ReportServiceUrl** and **ReportPath** property.

```
'js  
<div style="height: 600px; width: 100%; min-height: 404px;">  
    <Bold:ReportViewer runat="server" ID="viewer"  
        ReportServiceUrl="https://on-premise-demo.boldreports.com/reporting/reportservice/api/Viewer"  
        ReportPath="/Sample Reports/Company Sales"  
        OnClientAjaxBeforeLoad="onAjaxRequest">  
    </Bold:ReportViewer>  
</div>  
<script type="text/javascript">  
    function onAjaxRequest(args) {  
        args.headers.push({ Key: 'serverurl', Value: 'https://on-premise-  
            demo.boldreports.com/reporting/api/site/site1' });  
    }  
</script>  
`
```

You can also load the report using GUID instead of report location. Set the GUID of the report in the **ReportPath** property as **ReportPath= “91f24bf1-e537-4488-b19f-b37f77481d00”**.

Render RDLC report

The data binding support allows you to view RDLC reports that exist on the local file system with JSON array and custom business object data collection. The following steps demonstrates how to render an RDLC report with JSON array and custom business object data collection.

Add the RDLC report `product-list.rdlc` from Bold Reports installation location to your application Resources folder. For more information, see [Samples and demos](#).

Bind data source at client

Set the `ReportPath` and `ReportServiceUrl` to the report viewer initialization.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath("~/Resources/product-list.rdlc">
</Bold:ReportViewer>
</div>
`
```

The following code is used to set the report datasource client side in your application's `Page_Load()` function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
    ProductList productList = new ProductList();
    List<BoldReports.Models.ReportViewer.ReportDataSource> dataSources = new
    List<BoldReports.Models.ReportViewer.ReportDataSource>();
    BoldReports.Models.ReportViewer.ReportDataSource dataSource = new
    BoldReports.Models.ReportViewer.ReportDataSource();
    dataSource.Name = "list";
    dataSource.Value = productList.GetData();
    dataSources.Add(dataSource);
    this.viewer.DataSources = dataSources;
}
`
```

Bind data source in Web API controller

The following steps help you to configure the Web API to render the RDLC report with business object data collection.

Create a class and methods that returns business object data collection. Use the following code in your application Web API Service.

```
'csharp
[Serializable]
public class ProductList
{
    public string ProductName { get; set; }
    public string OrderId { get; set; }
    public double Price { get; set; }
    public string Category { get; set; }
    public string Ingredients { get; set; }
    public string ProductImage { get; set; }

    public static IList GetData()
    {
        List<ProductList> datas = new List<ProductList>();
        ProductList data = null;
        data = new ProductList()
        {
            ProductName = "Baked Chicken and Cheese",
            OrderId = "323B60",
            Price = 55,
            Category = "Non-Veg",
            Ingredients = "grilled chicken, corn and olives.",
            ProductImage = ""
        };
        datas.Add(data);
        data = new ProductList()
        {
            ProductName = "Chicken Delite",
            OrderId = "323B61",
        };
        datas.Add(data);
    }
}
```

```

Price = 100,
Category = "Non-Veg",
Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
    ProductName = "Chicken Tikka",
    OrderId = "323B62",
    Price = 64,
    Category = "Non-Veg",
    Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
    ProductImage = ""
};
datas.Add(data);
return datas;
}
}
`
```

Set the value of the `ProcessingMode` property to `ProcessingMode.Local` in the RDLC report location.

To set the `ReportPath` of the report by using the `MapPath`.

Bind the business object data values collection by adding a new item to the `DataSources` as in the following code snippet.

```

`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
    reportOption.ReportModel.ProcessingMode = ProcessingMode.Local;
    reportOption.ReportModel.ReportPath =
        System.Web.Hosting.HostingEnvironment.MapPath("~/Resources/product-list.rdlc");
    reportOption.ReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name = "list",
        Value = ProductList.GetData() });
}
```

Render subreport

Change subreport path

}

Here, the **Name** is case sensitive and it should be same as in the data source name in the report definition.

The **Value** accepts **IList**, **DataSet**, and **DataTable** inputs.

In the previous code, the **product-list.rdlc** report is loaded from the **Resources** folder location.

Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

Add the sub report and main reports to the application **Resources** folder. In this tutorial, the already created reports are used. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section.

Download the **side-by-side-main-report.rdl**, **side-by-side-sub-report.rdl** reports from [here](#). You can add a report from the Syncfusion installation location. For more information, refer to [Samples and demos](#).

Set the **ReportPath** and **ReportServiceUrl** properties of the Report Viewer as in the following code snippet.

The following code example demonstrates how to load a subreport in the Report Viewer at client side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer">
ReportPath="~/Resources/side-by-side-main-report.rdl"
</Bold:ReportViewer>
</div>
'
```

Change subreport path

To change the subreport file path, set the **Stream** property of **SubReportModel** in the **OnInitReportOptions** method.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
if (reportOption.SubReportModel != null)
```

Render subreport

Set subreport parameter

```
{  
FileStream SubStream = new  
FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/sub-report-detail.rdl"),  
FileMode.Open, FileAccess.Read);  
reportOption.SubReportModel.Stream = SubStream;  
}  
}  
`
```

Set subreport parameter

You can change the parameter default values of a subreport in the **OnReportLoaded** method of the Web API Controller as given in the following code snippet.

```
'csharp  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
if (reportOption.SubReportModel != null)  
{  
reportOption.SubReportModel.Parameters = new BoldReports.Web.ReportParameterInfoCollection();  
reportOption.SubReportModel.Parameters.Add(new BoldReports.Web.ReportParameterInfo()  
{  
Name = "SalesPersonID",  
Values = new List<string>() { "2" }  
});  
}  
}  
`
```

Modify subreport data source connection string

You can change the credential and connection information of the data sources used in the subreport using the **SubReportModel** in the **OnInitReportOptions** method.

```
'csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
if (reportOption.SubReportModel != null)  
{  
reportOption.SubReportModel.DataSourceCredentials = new  
List<BoldReports.Web.DataSourceCredentials>();
```

Render subreport

Set subreport data source

```
reportOption.SubReportModel.DataSourceCredentials.Add(new  
BoldReports.Web.DataSourceCredentials("NorthWind", "Data  
Source=dataplatformdemodata.syncfusion.com;Initial Catalog=Northwind;user id=demoreadonly@data-  
platform-demo;password=N@c=Y8s*1&dh"));  
}  
}  
`
```

Set subreport data source

You can bind local business object data source collection only for RDLC reports. To specify data source of a RDLC subreport, set the **ReportDataSource** property in the **OnReportLoaded** method.

The RDL report has the connection information in report definition itself, so no need to bind data source.

Add the RDLC sub report and main reports to your application **Resources** folder. You can download it from [here](#).

Set the **ReportPath** and **ReportServiceUrl** properties of the Report Viewer as shown in following code snippet.

```
`js  
<div style="height: 650px; width: 950px; min-height: 404px;">  
<Bold:ReportViewer runat="server" ID="viewer"  
ReportServiceUrl="/api/ReportViewer">  
ReportPath="~/Resources/product-list-main.rdlc"  
ProcessingMode="Local"  
</Bold:ReportViewer>  
</div>  
`
```

Create a class and methods that returns business object data collection. Use the following code in the application Web API Service.

```
`csharp  
public class ProductList  
{  
    public string ProductName { get; set; }  
    public string OrderId { get; set; }  
    public double Price { get; set; }  
    public string Category { get; set; }  
}
```

Render subreport

Set subreport data source

```
public string Ingredients { get; set; }

public string ProductImage { get; set; }

public static IList GetData()

{

List<ProductList> datas = new List<ProductList>();

ProductList data = null;

data = new ProductList()

{

ProductName = "Baked Chicken and Cheese",

OrderId = "323B60",

Price = 55,

Category = "Non-Veg",

Ingredients = "grilled chicken, corn and olives.",

ProductImage = ""

};

datas.Add(data);

data = new ProductList()

{

ProductName = "Chicken Delite",

OrderId = "323B61",

Price = 100,

Category = "Non-Veg",

Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",

ProductImage = ""

};

datas.Add(data);

data = new ProductList()

{

ProductName = "Chicken Tikka",

OrderId = "323B62",

Price = 64,

Category = "Non-Veg",

Ingredients = "onions, grilled chicken, chicken salami & tomatoes."}
```

Render subreport

Load subreport stream

```
ProductImage = ""  
};  
datas.Add(data);  
return datas;  
}  
}  
,
```

Bind the business object data values collection to the subreport by adding a new item to the **DataSources** as shown in the following code snippet.

```
`csharp  
public void OnReportLoaded(ReportViewerOptions reportOption)  
{  
    //Assigning the data source for 'product-list.rdlc'  
    if (reportOption.SubReportModel != null)  
    {  
        reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();  
        reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =  
            "list", Value = ProductList.GetData() });  
    }  
}
```

The data source name is case sensitive, it should be same as in the report definition.

Load subreport stream

To load subreport as stream, set the **Stream** property in the **OnInitReportOptions** method.

```
`csharp  
public void OnInitReportOptions(ReportViewerOptions reportOption)  
{  
    if (reportOption.SubReportModel != null)  
    {  
        FileStream reportStream = new  
        FileStream(System.Web.Hosting.HostingEnvironment.MapPath(@"~/Resources/product-list.rdlc"),  
        FileMode.Open, FileAccess.Read);  
        reportOption.SubReportModel.Stream = reportStream;  
    }  
}
```

| | |
|-------------------|-------------------------|
| Report parameters | Set parameter at client |
|-------------------|-------------------------|

```
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
//Assigning the data source for 'product-list.rdlc'
if (reportOption.SubReportModel != null)
{
reportOption.SubReportModel.DataSources = new BoldReports.Web.ReportDataSourceCollection();
reportOption.SubReportModel.DataSources.Add(new BoldReports.Web.ReportDataSource { Name =
"list", Value = ProductList.GetData() });

}
}
`
```

Report parameters

Provides property options to pass or set report parameters default values at run time using the `parameters` property. You can set the report parameters while creating the Report Viewer control in a script or in the Web API Controller.

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#).

Set parameter at client

Set the `ReportPath` and `ReportServiceUrl` to the report viewer initialization.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
`
```

The following code is used to set the report parameter client side in your application's `Page_Load()` function.

```
`csharp
protected void Page_Load(object sender, EventArgs e)
{
```

```
this.viewer.Parameters.Add(new BoldReports.Models.ReportViewer.ReportParameter()
{
    Name = "SalesOrderNumber",
    Values = new List<string>() { "SO50756" }
});
}
`
```

Set parameters in Web API Controller

To set parameter default value in Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    List<BoldReports.Web.ReportParameter> userParameters = new
    List<BoldReports.Web.ReportParameter>();
    userParameters.Add(new BoldReports.Web.ReportParameter()
    {
        Name = "SalesOrderNumber",
        Values = new List<string>() { "SO50756" }
    });
    reportOption.ReportModel.Parameters = userParameters;
}
`
```

Get report parameter

The **ReportHelper** class provides methods to get the report parameters used in the report. The following helper methods used to get parameter with or without values.

[Methods | Description](#)

[GetParameters](#) | Returns the parameters used in the current report without the processed values.

[GetParametersWithValues](#) | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

```
'csharp
public class ReportViewerController : ApiController, IReportController
{
    Dictionary<string, object> jsonArray = null;
```

Report interaction events

Get report parameter

```
public object PostReportAction(Dictionary<string, object> jsonResult)
{
    jsonArray = jsonResult;
    return ReportHelper.ProcessReport(jsonResult, this);
}

....
```

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
    var reportParameters = ReportHelper.GetParameters(jsonArray, this);
    List<BoldReports.Web.ReportParameter> setParameters = new
    List<BoldReports.Web.ReportParameter>();
    if (reportParameters != null)
    {
        foreach (var rptParameter in reportParameters)
        {
            setParameters.Add(new BoldReports.Web.ReportParameter()
            {
                Name = rptParameter.Name,
                Values = new List<string>() { "SO50756" }
            });
        }
        reportOption.ReportModel.Parameters = setParameters;
    }
}
```

,

Report interaction events

You can handle the report interaction events with reports using the following events.

ReportLoaded

ReportError

ShowError

Drill through

Hyperlink

Report loaded

The **OnClientReportLoaded** event occurs after the report is loaded and it is ready to start the processing. You can handle the event and specify the data source and parameters at client side. The following sample code loads a report by assigning the report data source input in the **OnClientReportLoaded** event.

In this tutorial, the **product-list.rdlc** report is used. You can add the report from the Bold Reports installation location. For more information, refer to [Samples and demos](#).

```
'js
<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">
<link href="Content/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="Scripts/jquery-3.3.1.min.js"></script>
<script src="Scripts/common/bold.reports.common.min.js"></script>
<script src="Scripts/common/bold.reports.widgets.min.js"></script>
<script src="Scripts/data-visualization/ej.chart.min.js"></script>
<script src="Scripts/bold.report-viewer.min.js"></script>
<div style="height: 600px; width: 100%; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ProcessingMode="Local"
ReportPath("~/Resources/product-list.rdlc"
OnClientReportLoaded="onReportLoaded">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onReportLoaded(args) {
var dataSource = [
{
ProductName: "Baked Chicken and Cheese", OrderId: "323B60", Price: 55, Category: "Non-Veg",
Ingredients: "Grilled chicken, Corn and Olives.", ProductImage: ""
},
{

```

```

ProductName: "Chicken Delite", OrderId: "323B61", Price: 100, Category: "Non-Veg", Ingredients:
"Cheese, Chicken chunks, Onions & Pineapple chunks.", ProductImage: ""

},
{

ProductName: "Chicken Tikka", OrderId: "323B62", Price: 64, Category: "Non-Veg", Ingredients: "Onions,
Grilled chicken, Chicken salami & Tomatoes.", ProductImage: ""

}];

var reportObj = $('#MainContent_viewer').data("boldReportViewer");
reportObj.model.dataSources = [{

value: ej.DataManager(dataSource).executeLocal(ej.Query()),

name: "list"

}];

}

</script>

</asp:Content>
`
```

Report error

When an error occurs in the report processing, it raises the `OnClientReportError` event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```

`js

<div style="height: 600px; width: 100%; min-height: 404px;">

<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ProcessingMode="Local"
<%--ReportPath="~/Resources/product-list.rdlc"--%>
OnClientReportError="onReportError">
</Bold:ReportViewer>
</div>

<script type="text/javascript">
function onReportError(args) {
alert(argserrmsg);
args.cancel = true;
}
</script>
```

To suppress the default error dialog, set the cancel argument to true.

[Show error](#)

The **OnClientShowError** event is invoked whenever users click a report item that contains an error in processing. It provides detailed information about the cause of error. You can hide the default dialog and show your customized dialog.

'js

```
<div style="height: 600px; width: 100%; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ProcessingMode="Local"
<%--ReportPath="~/Resources/product-list.rdlc"--%>
OnClientShowError="onShowError">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onShowError(args) {
    alert("Error code : " + args.errorCode + "\n" +
    "Error Detail : " + args.errorDetail + "\n" +
    "Error Message : " + args.errorMessage);
    args.cancel = true;
}
</script>
```

[Drill through](#)

When a drill through item is selected in a report, it invokes the **OnClientDrillThrough** event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

'js

```
<div style="height: 600px; width: 100%; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-person-details.rdl"
OnClientDrillThrough="onDrillThrough">
```

Handle post actions

Hyperlink

```
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onDrillThrough(args) {
args.actionInfo.ReportName = "personal-details";
args.actionInfo.Parameters = [{ name: 'EmployeeID', value: ['3'] }];
}
</script>
`
```

Hyperlink

The **OnClientHyperlink** event occurs when users click a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
`js
<div style="height: 600px; width: 100%; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/customer-support-analysis.rdl"
OnClientHyperlink="onHyperlink">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onHyperlink(args) {
args.cancel = true;
//You can modify the URL here
window.open(args.actionInfo.Hyperlink);
}
</script>
`
```

Handle post actions

Report processing actions are sent in an Ajax request to exchange data with the Web API service. You can handle post actions event to customize the Ajax requests.

AjaxBeforeLoad

AjaxSuccess**AjaxError**

AjaxBeforeLoad

This event can be triggered before an Ajax request is sent to the Report Viewer Web API service. It allows you to set additional headers and custom data in the Ajax request. The following code sample demonstrates adding custom authorization header and passing default parameter values to service.

Add custom header to Ajax request

Initialize the `OnClientAjaxBeforeLoad` event in the script and add the authorization token to the `headers` property.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
OnClientAjaxBeforeLoad="onAjaxRequest">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
}
</script>
`
```

In this tutorial, the `sales-order-detail.rdl` report is used, and it can be downloaded from [here](#).

Get the custom header value from the `HttpContext` header collection using the key name specified at client side.

```
'csharp
string authenticationHeader;
public object PostReportAction(Dictionary<string, object> jsonResult)
{
if (jsonResult != null)
{
//Get client side custom ajax header and store in local variable
authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
```

```
//Perform your custom validation here
if (authenticationHeader == "") {
{
return new Exception("Authentication failed!!!!");
}
else
{
return ReportHelper.ProcessReport(jsonResult, this);
}
}
return null;
}
`
```

Perform your own action to validate the header values.

Pass custom data in Ajax request

Use the `data` property to set custom data to the server in the Ajax request. In the following code sample, parameter values are passed to the server side.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
OnClientAjaxBeforeLoad="onAjaxRequest">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
</script>
`
```

The custom data values are stored in the `customData` header key, you can store it to the local property. The following code sample stores parameter values, then the values are set to the report in the `OnReportLoaded` method.

'csharp

```
public string DefaultParameter = null;
string authenticationHeader;
public object PostReportAction(Dictionary<string, object> jsonResult)
{
if (jsonResult != null)
{
if (jsonResult.ContainsKey("customData"))
{
//Get client side custom data and store in local variable. Here parameter values are sent.
DefaultParameter = jsonResult["customData"].ToString();
}

//Get client side custom ajax header and store in local variable
authenticationHeader = HttpContext.Current.Request.Headers["Authorization"];
//Perform your custom validation here
if (authenticationHeader == "")
{
return new Exception("Authentication failed!!!");
}
else
{
return ReportHelper.ProcessReport(jsonResult, this);
}
}
return null;
}

public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (DefaultParameter != null)
{
//Set client side custom header data
}
}
```

```
reportOption.ReportModel.Parameters =
JsonConvert.DeserializeObject<List<BoldReports.Web.ReportParameter>>(DefaultParameter);
}
}
`
```

AjaxSuccess

To perform custom action or show user defined message, use the **OnClientAjaxSuccess** event on the successful Ajax request.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
OnClientAjaxBeforeLoad="onAjaxRequest"
OnClientAjaxSuccess="onAjaxSuccess">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
function onAjaxSuccess(args) {
//Perform your custom success message here
alert("Ajax request success!!!!");
}
</script>
`
```

AjaxError

The **OnClientAjaxError** event is called, if an error occurred with the request, you can display the customized error detail in the event method.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
```

```
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
OnClientAjaxBeforeLoad="onAjaxRequest"
OnClientAjaxError="onAjaxFailure">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onAjaxRequest(args) {
args.headers.push({ Key: 'Authorization', Value: btoa('guest', 'demo@123') });
//Passing custom parameter data to server
args.data = [{ name: 'SalesOrderNumber', labels: ['SO50756'], values: ['SO50756'] }];
}
function onAjaxFailure(args) {
alert("Status: " + args.status + "\n" +
"Error: " + args.responseText);
}
</script>
`
```

You can never have both an error and a success callback with a request.

Error logging in ASP.NET Web Forms Report Viewer

If an error occurred in report processing, ASP.NET Web Forms Report Viewer displays short messages about the error in view. You need to configure the `ApiController` to save all logs, stack trace, and error information.

This section explains how to log the detailed error information to your ASP.NET Web Forms application.

This section requires a ASP.NET Web Forms Report Viewer application, if you don't have, then create by referring to the [Getting-Started](#) documentation.

In Solution Explorer, open the Report Viewer Controller file.

Inherit the `IReportLogger` interface and implement the interface methods.

```
'csharp
public class ReportViewerController : ApiController, IReportController, IReportLogger
{
```

```
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType errorType)
{
    throw new NotImplementedException();
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string methodName, string className)
{
    throw new NotImplementedException();
}

```

```

Create a method in `ReportViewerController` to write the error text into application folder.

```
`csharp
internal void WriteLogs(string errorMessage)
{
 string filePath = HttpContext.Current.Server.MapPath("/App_Data/Errordetails.txt");
 using (StreamWriter writer = new StreamWriter(filePath, true))
 {
 writer.AutoFlush = true;
 writer.WriteLine(errorMessage);
 }
}
`
```

Invoke the newly created function in `.LogError` as follows.

```
`csharp
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType errorType)
{
 WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string methodName, string className)
```

```
{
 WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:
 {3}", className, methodName, errorDetail, exception.StackTrace));
}
`
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final controller is given as follows, you can replace it in your application.

```
`csharp
public class ReportViewerController : ApiController, IReportController, IReportLogger
{
 // Post action for processing the RDL/RDLC report
 public object PostReportAction(Dictionary<string, object> jsonResult)
 {
 return ReportHelper.ProcessReport(jsonResult, this);
 }
 // Get action for getting resources from the report
 [System.Web.Http.ActionName("GetResource")]
 [AcceptVerbs("GET")]
 public object GetResource(string key, string resourcetype, bool isPrint)
 {
 return ReportHelper.GetResource(key, resourcetype, isPrint);
 }
 // Method that will be called when initialize the report options before start processing the report
 public void OnInitReportOptions(ReportViewerOptions reportOption)
 {
 // You can update report options here
 }
 // Method that will be called when reported is loaded
 public void OnReportLoaded(ReportViewerOptions reportOption)
 {
 // You can update report options here
 }
}
```

[Print report](#)[View report in print mode](#)

```
public void LogError(string message, Exception exception, MethodBase methodType, ErrorType
errorType)
{
 WriteLogs(string.Format("Error Message: {0} \n Stack Trace: {1}", message, exception.StackTrace));
}

public void LogError(string errorCode, string message, Exception exception, string errorDetail, string
methodName, string className)
{
 WriteLogs(string.Format("Class Name: {0} \n Method Name: {1} \n Error Message: {2} \n Stack Trace:
{3}", className, methodName, errorDetail, exception.StackTrace));
}

internal void WriteLogs(string errorMessage)
{
 // Error details text file path location
 string filePath = HttpContext.Current.Server.MapPath("/App_Data/Errordetails.txt");
 using (StreamWriter writer = new StreamWriter(filePath, true))
 {
 writer.AutoFlush = true;
 writer.WriteLine(errorMessage);
 }
}
}
```

## Print report

The Report Viewer provides print report option in the toolbar to print a copy of the report. The page setup dialog allows you to set the paper size or other page setup properties. To see print margins, click **Print Layout** on the toolbar.

You can set values in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values of the Page Setup dialog is based on the report properties set in the design view.

## [View report in print mode](#)

Print margins are displayed in the print layout only. To view report in print mode by default, set the **PrintMode** property to true.

```
`js
```

```
<div style="height: 650px; width: 950px; min-height: 404px;">
```

```
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
PrintMode="true">
</Bold:ReportViewer>
</div>
'
```

By default, the Report Viewer renders report in normal layout in which the print margins are not displayed.

### [Print in new page](#)

To open the print in a new tab of the current browser, set the `PrintOption` property to `NewTab`. By default, it shows the print dialog in the same page.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
PrintMode="true"
PrintOption="NewTab">
</Bold:ReportViewer>
</div>
'
```

The pop-up blocker should be enabled for the page to open the print view in new tab.

### [Set page orientation and paper size](#)

You can specify the print page paper size and orientation at client-side to change the page setup properties by setting the `page-settings` property.

The following code example illustrates how to set page orientation and paper size in the Report Viewer for client side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
```

```
</div>
```

```
,
```

The following code example illustrates how to set page orientation and paper size in your application's `Page_Load()` function.

```
'csharp
```

```
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.PageSettings = new BoldReports.Models.ReportViewer.PageSettings();
 this.viewer.PageSettings.Orientation = BoldReports.ReportViewerEnums.Orientation.Landscape;
 this.viewer.PageSettings.PaperSize = BoldReports.ReportViewerEnums.PaperSize.Letter;
}
```

```
,
```

### Set report margin

To set margin values to the report page setup, use the `Margins` property and specify the value to top, right, bottom, and left.

The following code example illustrates how to set report margin in the Report Viewer for client side.

```
'js
```

```
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
```

```
,
```

The following code example illustrates how to set report margin in your application's `Page_Load()` function.

```
'csharp
```

```
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.PageSettings = new BoldReports.Models.ReportViewer.PageSettings();
 this.viewer.PageSettings.Margins = new BoldReports.Models.ReportViewer.Margins();
 this.viewer.PageSettings.Margins.Top = 0.5;
```

[Print report](#)

[Set page height and width](#)

```
this.viewer.PageSettings.Margins.Bottom = 0.5;
this.viewer.PageSettings.Margins.Right = 0.5;
this.viewer.PageSettings.Margins.Left = 0.5;
}
,
```

The values set in the margin property is considered as inches input.

### [Set page height and width](#)

To set height and width values to the report page setup, use the **Height** and **Width** properties.

The following code example illustrates how to set page height and width in the Report Viewer for client side.

```
'js
<div style="height: 650px;width: 950px;min-height:404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
,
```

The following code example illustrates how to set page height and width in your application's **Page\_Load()** function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.PageSettings = new BoldReports.Models.ReportViewer.PageSettings();
 this.viewer.PageSettings.Height = 11.69;
 this.viewer.PageSettings.Width = 8.27;
}
```

The values set in the height and width properties are considered as inches input.

### [Print report with images](#)

When the report has more images, the browser will send the report stream to the print dialog before the images are completely loaded. To load the print report stream with complete images, you should set the **EmbedImageData** property to true in **OnInitReportOptions** as shown in the following code.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.EmbedImageData = true;
}
`
```

Replace the following code sample in the client-side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/product-details.rdl">
</Bold:ReportViewer>
</div>
`
```

In this tutorial, the `product-details.rdl` report is used, and it can be downloaded from [here](#).

### External styles in report printing

While printing report, the external styles are used in the application overrides printable page style and prints output with incorrect alignments. To avoid the external script overriding, set the `isStyleLoad` property to false, which will print the page using only the Report Viewer styles.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/product-details.rdl"
OnClientReportPrint="onReportPrint">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onReportPrint(args) {
 args.isStyleLoad = false;
}
</script>
`
```

## Show print progress

Report Viewer provides events to show the progress information when the printing takes long time to complete.

To show print progress, follow these steps:

Set the `OnClientPrintProgressChanged` in Report Viewer initialization.

Implement the function and add code samples to show a custom message based on the print progress status as shown in the following code snippet.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath("~/Resources/product-details.rdl"
OnClientPrintProgressChanged="onPrintProgressChanged">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onPrintProgressChanged(args) {
if (args.stage == "beginPrint") {
$('#MainContent_viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing
print data.. Please wait..." });
}
if (args.stage == "printStarted") {
var popupObj = $('#MainContent_viewer').data('ejWaitingPopup');
popupObj.hide();
}
else if (args.stage == "preparation") {
console.log(args.stage);
if (args.preparationStage == "dataPreparation") {
console.log(args.preparationStage);
console.log(args.totalPages);
console.log(args.currentPage);
if (args.totalPages > 1 && args.currentPage > 1) {
var progressPercentage = Math.floor((args.currentPage / args.totalPages) * 100);
}
```

```

if (progressPercentage > 0) {
 var popupObj = $('#MainContent_viewer').data('ejWaitingPopup');
 popupObj.setModel({ text: "Preparing print data.." + progressPercentage + " % completed.. Please wait..." });
}
}
}
}
}

args.handled = true;
}

</script>
`
```

### [Remove empty spaces in printing](#)

The extra blank page is created when the body of your report is too wide for your page. To make the report appear on a single page, all the content within the report body must fit on the physical page, and the body width should be as the following formula.

Body Width <= Page Width - (Left Margin + Right Margin)

For more details about removing the empty pages in the report while designing, refer to the knowledge base article of [report page sizing](#).

## Export report

The Report Viewer provides events and properties to control and customize the report exporting functionality.

### [Export event handling](#)

You can show the progress information, when the exporting process takes long time to complete using the `OnClientExportProgressChanged` event.

Set the `OnClientExportProgressChanged` event in Report Viewer initialization.

Implement the function and replace the following code samples to show a custom message based on the progress stage.

The following code example demonstrates how to export event handling in the Report Viewer at client side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
```

```
ReportPath="~/Resources/sales-order-detail.rdl"
OnClientExportProgressChanged="onExportProgressChanged">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onExportProgressChanged(args) {
if (args.stage === "beginExport") {
console.log(args.stage);
args.format =
$('#MainContent_viewer').ejWaitingPopup({ showOnInit: true, cssClass: "customStyle", text: "Preparing
exporting document.. Please wait..." });
}
else if (args.stage === "exportStarted") {
console.log(args.stage);
var popupObj1 = $('#MainContent_viewer').data('ejWaitingPopup');
popupObj1.hide();
}
else if (args.stage === "preparation") {
console.log(args.stage);
console.log(args.format);
console.log(args.preparationStage);
if (args.format === "PDF" && args.preparationStage === "documentPreparation") {
console.log(args.totalPages);
console.log(args.currentPage);
if (args.totalPages > 1 && args.currentPage > 1) {
var progressPercentage = Math.floor((args.currentPage / args.totalPages) * 100);
if (progressPercentage > 0) {
var popupObj2 = $('#MainContent_viewer').data('ejWaitingPopup');
popupObj2.setModel({ text: "Preparing exporting document.." + progressPercentage + " % completed..
Please wait..." });
}
}
}
}
}
}
```

Export report

Export data visualization items

```
args.handled = true;
}
</script>
'
```

### Export data visualization items

To export the reports with data visualization components, it is mandatory to configure the web scripts in Report Viewer Web API controller. If the report definition uses chart, gauge and map report items then configure the scripts in Web API as the following steps,

Open the Report Viewer Web API controller.

Configure the below scripts and styles in **OnInitReportOptions** on Web API controller.

jquery-1.10.2.min.js

bold.reports.common.min.js

bold.reports.widgets.min.js

ej.chart.min.js

ej2-base.min.js

ej2-data.min.js

ej2-pdf-export.min.js

ej2-svg-base.min.js

ej2-lineargauge.min.js

ej2-circulargauge.min.js

ej.map.min.js

bold.report-viewer.min.js

You can replace the following codes in Report Viewer Web API controller.

```
`csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
```

```
{
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
 //Chart component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
 //Gauge component scripts
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
 //Map component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
 //Report Viewer Script
 "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
};
reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
 "https://code.jquery.com/jquery-1.10.2.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

#### Export data visualization items in azure environment

Report Viewer uses **WebBrowser** to export the data visualization items to PDF, Word, Excel file formats. The **WebBrowser** is not supported in azure environment. To overcome this limitation in azure environment, we have provided an option to export the data visualization report items using [PhantomJS](#).

To download **PhantomJS** application and deploy it on your machine, you should accept its license terms on [LICENSE](#) and [Third-Party](#) document.

Download **PhantomJS** from [here](#) and extract the download file.

Copy the **PhantomJS.exe** file from the extracted bin folder and paste into **PhantomJS** folder in your application.

Open the Report Viewer Web API controller.

Set the `UsePhantomJS` property to true and `PhantomJSPath` property in `OnInitReportOptions` method.

```
'csharp
```

```
// Method will be called to initialize the report information to load the report with ReportHelper for processing.
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.ExportResources.UsePhantomJS = true;
 reportOption.ReportModel.ExportResources.PhantomJSPath = @"\PhantomJS\";
 reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
 {
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
 //Chart component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
 //Gauge component scripts
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
 //Map component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
 //Report Viewer Script
 "https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
 };
 reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
 {
 "https://code.jquery.com/jquery-1.10.2.min.js"
 };
}
```

The Scripts and Dependent scripts must be added to export the items. For more details refer to the [export data visualization items](#) section.

### Change Excel and Word export format

Allows you change the default file format to any other file format using the ExcelFormat and WordFormat properties.

The following code example demonstrates how to change Excel and Word export format in the Report Viewer at client side.

```
'js
<div style="height: 650px;width: 950px;min-height:404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
'
```

The following code example demonstrates how to change Excel and Word export format in your application's Page\_Load() function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.ExportSettings = new BoldReports.Models.ReportViewer.ExportSettings();
 this.viewer.ExportSettings.ExcelFormat = BoldReports.ReportViewerEnums.ExcelFormats.Excel2013;
 this.viewer.ExportSettings.WordFormat = BoldReports.ReportViewerEnums.WordFormats.Word2013;
}
```

### Hide specific export type for report

Show or hide the default export types available in the component using the ExportOptions property.

The following code example demonstrates how to hide specific export type to the report in the Report Viewer at client side.

```
'js
<div style="height: 650px;width: 950px;min-height:404px;">
```

```
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
'
```

The following code example demonstrates how to hide specific export type to the report in your application's `Page_Load()` function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.ExportSettings = new BoldReports.Models.ReportViewer.ExportSettings();
 this.viewer.ExportSettings.ExportOptions = BoldReports.ReportViewerEnums.ExportOptions.All &
 ~BoldReports.ReportViewerEnums.ExportOptions.Html;
}
'
```

## PDF export options

The `PDFOptions` provides properties to manage PDF export behaviors. You should set the properties in the `OnInitReportOptions` method of the Web API service.

### Export with complex scripts

To export reports with the complex scripts, set the `EnableComplexScript` property of `PDFOptions` instance to true.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
 {
 EnableComplexScript = true
 };
}
```

## PDF conformance

You can export the report as a PDF/A-1b document by specifying the `PdfConformanceLevel.Pdf_A1B` conformance level in the `PdfConformanceLevel` property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
 {
 PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B
 };
}
`
```

#### Add custom PDF fonts

You can add custom fonts to the PDF exported document by adding the font streams to **Fonts** collection in **PDFOptions** instance.

To add custom fonts to the PDF exported document, follow these steps:

Add the font **.ttf** files into your application **Resources** folder.

In the Solution Explorer, open the properties of the font file and set the **Copy** property to **Output Directory** as **Copy always**.

Initialize the **Font** collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 string basePath = _hostingEnvironment.WebRootPath;
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions()
 {
 //Load Missing font stream
 Fonts = new Dictionary<string, System.IO.Stream>
 {
 { "Segoe UI", new FileStream(basePath + @"\Resources\font_symbols.ttf", FileMode.Open,
 FileAccess.Read) }
 }
 };
}
```

If any fonts used in the report definition is not installed or available in the local system, then you should load the font stream. In the above code, `font_symbols` font stream is loaded to export the `sales-order-detail.rdl` report as symbols.

### Word export options

The `WordOptions` provides properties to manage Word document export behaviors.

#### Word document type

You can save the report to the required document version by setting the `FormatType` property.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
 {
 FormatType = BoldReports.Writer.WordFormatType.Docx
 };
}
```

#### Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the `LayoutOption` to `TopLevel`. The `ParagraphSpacing` is the distance value added between two elements in the document.

'csharp

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
 {
 LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel,
 ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()
 {
 Bottom = 0.5f,
 Top = 0.5f
 }
 };
}
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in the word document is not a stand-alone object. If you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, add an empty paragraph between two tables.

#### Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without password. The following are the types of protection:

**AllowOnlyComments:** Adds or modifies only the comments in the Word document.

**AllowOnlyFormFields:** Modifies the form field values in the Word document.

**AllowOnlyRevisions:** Accepts or rejects the revisions in the Word document.

**AllowOnlyReading:** Views the content only in the Word document.

**NoProtection:** Accesses or edits the Word document contents as normally.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
 {
 ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading
 };
}
```

#### Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

##### Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

```
'csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
 {
 ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013
 }
}
```

```
};
}
`
```

### Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the `LayoutOption` to `IgnoreCellMerge`.

```
`csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
 {
 LayoutOption = BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge
 };
}
```

### Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the `ExcelSheetProtection` or enabling the `ReadOnlyRecommended` properties.

```
`csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
 {
 ReadOnlyRecommended = true,
 ExcelSheetProtection = Syncfusion.XlsIO.ExcelSheetProtection.DeletingColumns
 };
}
```

### CSV export options

The `CsvOptions` allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
`csharp
```

```
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 reportOption.ReportModel.CsvOptions = new BoldReports.Writer.CsvOptions()
```

Export report

Password protect exported document

```
{
 Encoding = System.Text.Encoding.Default,
 FieldDelimiter = "," ,
 UseFormattedValues = false,
 Qualifier = "#",
 RecordDelimiter = "@",
 SuppressLineBreaks = true,
 FileExtension = ".txt"
};
}
`
```

### [Password protect exported document](#)

Allows you protect the exported document such as PDF, Word, Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet illustrates how to encrypt the exported document with user-defined password.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 //PDF encryption
 reportOption.ReportModel.PDFOptions = new BoldReports.Writer.PDFOptions();
 reportOption.ReportModel.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
 {
 UserPassword = "password"
 };
 //Word encryption
 reportOption.ReportModel.WordOptions = new BoldReports.Writer.WordOptions()
 {
 EncryptionPassword = "password"
 };
 //Excel encryption
 reportOption.ReportModel.ExcelOptions = new BoldReports.Writer.ExcelOptions()
 {
 PasswordToModify = "password",
 PasswordToOpen = "password"
 };
```

```
};

//PPT encryption
reportOption.ReportModel.PPTOptions = new BoldReports.Writer.PPTOptions()
{
 EncryptionPassword = "password"
};

}

`
```

Password protection is not supported for HTML export format.

## Toolbar customization

You can hide the component toolbar to show customized user interface or to customize the toolbar icons and element's appearances using the templates and Report Viewer toolbar customization properties.

In this tutorial, the `sales-order-detail.rdl` report is used and it can be downloaded from [here](#). You can add the reports from the Syncfusion installation location. For more information, refer to [Samples and demos](#).

### Hide parameter block and toolbar items

To hide toolbar items, set the `ToolbarSettings` property. The following code can be used to remove the parameter option from the toolbar and hide the parameter block.

The following code example demonstrates how to hide the parameter block in the Report Viewer at client side.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
 <Bold:ReportViewer runat="server" ID="viewer"
 ReportServiceUrl="/api/ReportViewer"
 ReportPath="~/Resources/sales-order-detail.rdl">
 </Bold:ReportViewer>
</div>
`
```

The following code example demonstrates how to hide the parameter block in your application's `Page_Load()` function.

```
`csharp
protected void Page_Load(object sender, EventArgs e)
```

```
{
this.viewer.ToolbarSettings = new BoldReports.Models.ReportViewer.ToolbarSettings();
this.viewer.ToolbarSettings.Items = BoldReports.ReportViewerEnums.ToolbarItems.All &
~BoldReports.ReportViewerEnums.ToolbarItems.Parameters;
}
```

Similarly, you can show or hide all other toolbar options with the help of `ToolbarSettings.Items` enum.

### [Enable stop option in toolbar](#)

To enable stop option in toolbar, set the `ToolbarSettings.Items` property to `BoldReports.ReportViewerEnums.ToolbarItems.All`. The following code can be used to enable stop option in toolbar.

The following code example demonstrates how to enable stop option in the Report Viewer toolbar at client side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
'
```

The following code example demonstrates how to enable stop option in your application's `Page_Load()` function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
this.viewer.ToolbarSettings = new BoldReports.Models.ReportViewer.ToolbarSettings();
this.viewer.ToolbarSettings.Items = BoldReports.ReportViewerEnums.ToolbarItems.All;
}
```

### [Hide toolbar](#)

To hide the Report Viewer toolbar, set the `ShowToolbar` property to false.

The following code example demonstrates how to hide the toolbar in the Report Viewer at client side.

```
'js
<div style="height: 650px;width: 950px;min-height:404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
'
```

The following code example demonstrates how to hide the toolbar in your application's `Page_Load()` function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.ToolbarSettings = new BoldReports.Models.ReportViewer.ToolbarSettings();
 this.viewer.ToolbarSettings.ShowToolbar = false;
}
```

### Decide or hide the export option

The Report Viewer provides the `ExportOptions` property to show or hide the default export types available in the component. The following code hides the HTML export type from the default export options.

The following code example demonstrates how to decide or hide the export option in the Report Viewer at client side.

```
'js
<div style="height: 650px;width: 950px;min-height:404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl">
</Bold:ReportViewer>
</div>
'
```

The following code example demonstrates how to decide or hide the export option in your application's `Page_Load()` function.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 this.viewer.ExportSettings = new BoldReports.Models.ReportViewer.ExportSettings();
 this.viewer.ExportSettings.ExportOptions = BoldReports.ReportViewerEnums.ExportOptions.All &
 ~BoldReports.ReportViewerEnums.ExportOptions.Html;
}
`
```

### Add custom items to the export drop-down

To add custom items to the export drop-down available in the Report Viewer toolbar, use the property `CustomItems` and provide the JSON array of collection input with the `Index`, `CssClass` name, and `Value` properties. Register the `OnClientExportItemClick` event to handle the custom item actions as given in following code snippet.

You can use the following codes to add custom items to the export drop-down in your application's `Page_Load()` function using `ExportSettings` property.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 ExportSettings exportSettings = new ExportSettings();
 exportSettings.CustomItems = new List<CustomExportItem>();
 var exportItem1 = new CustomExportItem() { Index = 2, CssClass = "", Value = "Text File" };
 var exportItem2 = new CustomExportItem() { Index = 4, CssClass = "", Value = "DOT" };
 exportSettings.CustomItems.Add(exportItem1);
 exportSettings.CustomItems.Add(exportItem2);
 this.viewer.ExportSettings = exportSettings;
}
`
```

You can use the following codes to create an `OnClientExportItemClick` event at client side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
```

```
ReportPath="~/Resources/sales-order-detail.rdl" OnClientExportItemClick="onExportItemClick">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
//Export click event handler
function onExportItemClick(args) {
if (args.value === "Text File") {
//Implement the code to export report as Text
alert("Text File export option clicked");
} else if (args.value === "DOT") {
//Implement the code to export report as DOT
alert("DOT export option clicked");
}
}
</script>
`
```

### Add custom toolbar item

You can add custom items to Report Viewer toolbar using the `ToolbarSettings` property. You must register the `OnClientToolBarItemClick` event to handle the newly added custom items action.

You can use the following codes to add new toolbar group in your application's `Page_Load()` function using `ToolbarSettings` property.

```
'csharp
protected void Page_Load(object sender, EventArgs e)
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 toolbarSettings.CustomItems = new List<CustomItem>();
 var customItem = new CustomItem()
 {
 GroupIndex = 1,
 Index = 1,
 CssClass = "e-icon e-mail e-reportviewer-icon",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "E-Mail",
 Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
 }
}
```

```

};

toolbarSettings.CustomItems.Add(customItem);

this.viewer.ToolbarSettings = toolbarSettings;

}
`
```

You can use the following codes to create an `OnClientToolBarItemClick` event at client side.

```

`js

<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl" OnClientToolBarItemClick="onToolBarItemClick">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onToolBarItemClick(args) {
if (args.value == "E-Mail") {
alert('Action Triggered');
}
}
</script>
`
```

#### Add custom item to exiting toolbar group

To add a custom item to existing toolbar group use the property `CustomGroup` in `ToolbarSettings` and provide the JSON array of collection input with the `GroupIndex`, `Items`, `Type`, `CssClass` name, and `Tooltip` properties as given in following code snippet.

You can use the following codes to add custom item to exiting toolbar group in your application's `Page_Load()` function using `ToolbarSettings` property.

```

`csharp

protected void Page_Load(object sender, EventArgs e)
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 var groupItem = new List<CustomItem>();
 groupItem.Add(new CustomItem()
 {
```

```
CssClass = "e-icon e-mail e-reportviewer-icon CustomGroup",
Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
Id = "CustomGroup",
Tooltip = new ToolTip() { Header = "CustomGroup", Content = "toolbargroups" }
});
groupItem.Add(new CustomItem()
{
CssClass = "e-icon e-mail e-reportviewer-icon subCustomGroup",
Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
Id = "subCustomGroup",
Tooltip = new ToolTip() { Header = "subCustomGroup", Content = "subtoolbargroups" }
});
toolbarSettings.CustomGroups.Add(new CustomGroup() { Items = groupItem, GroupIndex = 4 });
this.viewer.ToolbarSettings = toolbarSettings;
}
`
```

You can use the following codes to create an **OnClientToolBarItemClick** event at client side.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl" OnClientToolBarItemClick="onToolBarItemClick">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onToolBarItemClick(args) {
if (args.value === "CustomGroup") {
//Implement the code to CustomGroup toolbar option
alert("CustomGroup toolbar option clicked");
}
if (args.value === "subCustomGroup") {
//Implement the code to subCustomGroup toolbar option
alert("SubCustomGroup toolbar option clicked");
}
```

```
}
```

```
}
```

```
</script>
```

```
,
```

### Add new toolbar group

To add new toolbar group and custom items to it, use the property `CustomItems` in `ToolbarSettings` and provide the JSON array of collection input with the `GroupIndex`, `Index` properties. The `CustomItems` must have the properties `Type`, `CssClass` and `Tooltip` as given in following code snippet.

You can use the following codes to add new toolbar group in your application's `Page_Load()` function using `ToolbarSettings` property.

```
'csharp
```

```
protected void Page_Load(object sender, EventArgs e)
```

```
{
```

```
 ToolbarSettings toolbarSettings = new ToolbarSettings();
```

```
 toolbarSettings.CustomItems = new List<CustomItem>();
```

```
 var customItem = new CustomItem()
```

```
 {
```

```
 GroupIndex = 1,
```

```
 Index = 1,
```

```
 CssClass = "e-icon e-mail e-reportviewer-icon",
```

```
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
```

```
 Id = "E-Mail",
```

```
 Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
```

```
 };
```

```
 toolbarSettings.CustomItems.Add(customItem);
```

```
 this.viewer.ToolbarSettings = toolbarSettings;
```

```
}
```

```
,
```

## Custom Actions

Add user defined buttons to the toolbar and invoke custom actions using the `Report Viewer` property. You can create a custom email button to send an email with the rendered report to users.

## Add email button

Create an email button in the toolbar using the `CustomItems` property with the values such as `groupIndex`, `index`, `itemType`, `cssClass`, and `tooltip`. The `OnClientToolBarItemClick` event triggers when you click the email button.

Access the Report Viewer model and create a JSON array for sending requests to the Web API Server. You can use the following codes to create an event with custom action.

You can use the following codes to add email button in your application's `Page_Load()` function using `ToolbarSettings` property.

`'csharp`

```
protected void Page_Load(object sender, EventArgs e)
{
 ToolbarSettings toolbarSettings = new ToolbarSettings();
 toolbarSettings.CustomItems = new List<CustomItem>();
 var customItem = new CustomItem()
 {
 GroupIndex = 1,
 Index = 1,
 CssClass = "e-icon e-mail e-reportviewer-icon",
 Type = BoldReports.ReportViewerEnums.ToolBarItemType.Default,
 Id = "E-Mail",
 Tooltip = new ToolTip() { Header = "E-Mail", Content = "Send rendered report as mail attachment" }
 };
 toolbarSettings.CustomItems.Add(customItem);
 this.viewer.ToolbarSettings = toolbarSettings;
}
```

You can use the following codes to create an `OnClientToolBarItemClick` event at client side.

```
'js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
```

```
ReportPath="~/Resources/sales-order-detail.rdl" OnClientToolBarItemClick="onToolBarItemClick">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onToolBarItemClick(args) {
 alert('Action Triggered');
}
</script>
`
```

You can use the following codes to invoke custom actions using custom email button at client side.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl" OnClientToolBarItemClick="onToolBarItemClick">
</Bold:ReportViewer>
</div>
<script type="text/javascript">
function onToolBarItemClick(args) {
if (args.value == "E-Mail") {
var proxy = $('#viewer').data('boldReportViewer');
var Report = proxy.model.reportPath;
var lastsIndex = Report.lastIndexOf("/");
var reportName = Report.substring(lastsIndex + 1);
var requrl = proxy.model.reportServiceUrl + '/SendEmail';
var _json = {
exportType: "PDF", reportViewerToken: proxy._reportViewerToken, ReportName: reportName
};
$.ajax({
type: "POST",
contentType: "application/json; charset=utf-8",
url: requrl,
```

```
data: JSON.stringify(_json),
dataType: "json",
crossDomain: true
})
}
}
</script>
`
```

## Create custom email action

Create a new action method `SendEmail` in the Web API service.

Export the report to the required type using the `ReportHelper.GetReport` method to send a report stream as an attachment.

The following code sample exports the report to stream and send it as an attachment to a specified mail address. In the code, the `SmtpClient` method is used to send the report as an email attachment.

```
`csharp
public object SendEmail(Dictionary<string, object> jsonResult)
{
 string _token = jsonResult["reportViewerToken"].ToString();
 var stream = ReportHelper.GetReport(_token, jsonResult["exportType"].ToString());
 stream.Position = 0;
 if (!ComposeEmail(stream, jsonResult["reportName"].ToString()))
 {
 return "Mail not sent !!!";
 }
 return "Mail Sent !!!";
}

public bool ComposeEmail(Stream stream, string reportName)
{
 try
 {
 MailMessage mail = new MailMessage();
```

```
SmtpClient SmtpServer = new SmtpClient("smtp.gmail.com");
mail.IsBodyHtml = true;
mail.From = new MailAddress("xx@gmail.com");
mail.To.Add("xx@gmail.com");
mail.Subject = "Report Name : " + reportName;
stream.Position = 0;
if (stream != null)
{
 ContentType ct = new ContentType();
 ct.Name = reportName + DateTime.Now.ToString() + ".pdf";
 System.Net.Mail.Attachment attachment = new System.Net.Mail.Attachment(stream, ct);
 mail.Attachments.Add(attachment);
}
SmtpServer.Port = 587;
SmtpServer.Credentials = new System.Net.NetworkCredential("xx@gmail.com", "xx");
SmtpServer.EnableSsl = true;
SmtpServer.Send(mail);
return true;
}
catch (Exception ex)
{
 throw ex;
}
return false;
}
`
```

In the above code sample, the report is exported to PDF format and send to users using the `SmtpClient` method.

## Localization of Bold Reports ASP.NET Web Forms Report Viewer

Localization of ASP.NET Web Forms Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. To render the static text with specific culture, refer to the following corresponding culture script files and set culture name to the `locale` property of the Report Viewer.

`ej.localetexts.fr-FR.min.js`

`ej.culture.fr-FR.min.js`

Install `BoldReports.Global` Nuget package in your ASP.NET application.

Refer to this `ej.localetexts.fr-FR.min.js` script file from the `Scripts` folder of your application.

'html

```
<script src="Scripts/bold-reports/I10n/reportdesigner/ej.localetexts.fr-FR.min.js"></script>
```

'

Refer to this `ej.culture.fr-FR.min.js` script file from the `Scripts` folder of your application.

'html

```
<script src="Scripts/bold-reports/i18n/ej.culture.fr-FR.min.js"></script>
```

'

To render the localization Report Viewer, use the following code snippet in your application.

'js

```
<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">
<link href="Content/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="Scripts/jquery-1.10.2.min.js"></script>
<script src="Scripts/common/bold.reports.common.min.js"></script>
<script src="Scripts/common/bold.reports.widgets.min.js"></script>
<script src="Scripts/data-visualization/ej.chart.min.js"></script>
<script src="Scripts/bold.report-viewer.min.js"></script>
<script src="Scripts/I10n/ej.localetexts.fr-FR.min.js"></script>
<script src="Scripts/i18n/ej.culture.fr-FR.min.js"></script>
<div style="height: 600px; width: 100%; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
locale="fr-FR">
</Bold:ReportViewer>
</div>
```

```
</asp:Content>
```

```
'
```

## Responsive layout rendering of ASP.NET Report Viewer

Report Viewer will adaptively render itself with optimal user interfaces for phone, tablet, or desktop form factors. This helps your application to scale elegantly on all form factors with ease. You can enable responsive layout rendering in Report Viewer by setting `isResponsive` property to true.

```
'js
```

```
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer"
ReportServiceUrl="/api/ReportViewer"
ReportPath="~/Resources/sales-order-detail.rdl"
IsResponsive="true">
</Bold:ReportViewer>
</div>
```

```
'
```

### Normal layout

The following output shows the normal layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in normal layout](/static/assets/aspnet-web-forms/report-viewer/images/responsive-layout/normal-layout.png)

### Responsive layout

The following output shows the responsive layout rendering of Report Viewer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/aspnet-web-forms/report-viewer/images/responsive-layout/responsive-layout.png)

## Limitations

### RDL specification

The Report Viewer control does not support RDL specification for SQL Server 2000 and SQL Server 2005.

### Report layout

Vertical alignment in the text box report item is not supported in web rendering.

In the Tablix cell split layout process, the entire cell moves to the next page to display the complete cell items, when the table cell width value exceeds the page width.

### Expressions

The object function and VB function do not have complete support.

## SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you should specify the data source credentials for each report data source to establish database connection.

## Samples and Demos

Browse and explore the ready-to-use RDL, RDLC reports, samples, online, and offline demos.

### Locally installed reports

You can obtain the sample RDL and RDLC files from the Bold Reports installed location

`%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.`

### Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

### Online demos

You can view the ASP.NET web forms Report Viewer online demo samples from [here](#).

### GitHub demo samples

Click [here](#) to view the GitHub Report Viewer demo samples.

## Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of ASP.NET Web Forms Report Viewer application:

### Client-side migration

In the Solution Explorer, right-click the **References** and remove the following assembly references:

`Syncfusion.EJ`

`Syncfusion.EJ.Web`

Add the assembly from the Bold Reports NuGet package `BoldReports.WebForms`. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for `BoldReports.WebForms` NuGet package, and install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

## Scripts and CSS references

Remove the following scripts and CSS references from the Report Viewer `Default.aspx` page.

`ej.web.all.min.css`

`ej.web.all.min.js`

The Report Viewer scripts and style sheets are added to the `Scripts` and `Content` folders in your application when installing the `BoldReports.WebForms` NuGet package. Add scripts as in the following code sample.

```
'html
```

```
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

## Adding data visualization scripts

To render the report with data visualization components such as chart, gauge, and map items, add scripts of the visualization element. The following table shows the script reference that needs to be added in Report Viewer page for data visualization elements.

Visualization item | Script file

Chart | `ej.chart.min.js`

Gauge | `ej2-base.min.js`, `ej2-data.min.js`, `ej2-pdf-export.min.js`, `ej2-svg-base.min.js`, `ej2-lineargauge.min.js` and `ej2-circulargauge.min.js`

Map | `ej.map.min.js`

To render the chart report item, add chart control script `ej.chart.min.js` before the `bold.report-viewer.min.js` reference in the `Default.aspx` page as demonstrated in the following code sample.

```
'html
```

```
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
```

```
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
```

```
\
```

The following code can be used to render the chart, gauge, and map report items in Report Viewer.

```
'html
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="Scripts/bold-reports/common/ej2-base.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-data.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-pdf-export.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-svg-base.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js"></script>
<!--Used to render the map item. Add this script only if your report contains the map report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
\
```

## Control initialization

The component prefix has been changed from `ej` to `Bold`.

Open the `Web.config` file and add the `BoldReports.WebForms` assembly reference to the `configuration` element with `Bold` tag prefix.

```
'html
<configuration>
....
....
<system.web>
....
```

```
<pages>
...
<controls>
<add assembly="BoldReports.WebForms" namespace="BoldReports.WebForms" tagPrefix="Bold" />
...
</controls>
</pages>
</system.web>
...
...
</configuration>
`
```

Open the Report Viewer component page.

Replace the tag `ej:ReportViewer` as `Bold:ReportViewer`.

```
`js
<div style="height: 650px; width: 950px; min-height: 404px;">
<Bold:ReportViewer runat="server" ID="viewer">
</Bold:ReportViewer>
</div>
`
```

If your application contains Report Viewer initialization in multiple pages then replace in all the pages.

## Server-side migration

In the Solution Explorer, right-click the **References** and remove the `Syncfusion.EJ.ReportViewer` assembly reference.

Add the assembly from the Bold Reports NuGet package `BoldReports.Web`. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for `BoldReports.Web` NuGet package, and then install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

## Web API Controller

The `IReportController` interface is moved to `BoldReports.Web.ReportViewer`. Open the Report Viewer Web API Controller file and remove the following using statement.

```
'csharp
using Syncfusion.EJ.ReportViewer;
'
```

Add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Your application is successfully upgraded to the latest version of Report Viewer, and you can run the application with new assemblies.

## Report export configuration

Now, the `BoldReports.Web` can export the reports with data visualization components only using web components. It is mandatory to configure the web scripts in Report Viewer Web API controller for exporting data visualization components such as chart, gauge, and map that are used in report definition. To configure the scripts in Web API, refer to the following steps:

Open the Report Viewer Web API controller.

Configure the following scripts and styles in `OnInitReportOptions` on Web API controller:

`jquery-1.10.2.min.js`

`bold.reports.common.min.js`

`bold.reports.widgets.min.js`

`ej.chart.min.js` - Exports the chart item. Add this script only if your report contains the chart report item.

`ej2-base.min.js`, `ej2-data.min.js`, `ej2-pdf-export.min.js`, `ej2-svg-base.min.js`, `ej2-lineargauge.min.js` and `ej2-circulargauge.min.js` - Exports the gauge item. Add this script only if your report contains the gauge report item.

`ej.map.min.js` - Exports the map item. Add this script only if your report contains the map report item.

`bold.report-viewer.min.js`

Replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
 reportOption.ReportModel.ExportResources.Scripts = new List<string>
 {
 resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
 resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
 //Chart component script
 resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
 //Gauge component scripts
 resourcesPath + @"\bold-reports\common\ej2-base.min.js",
 resourcesPath + @"\bold-reports\common\ej2-data.min.js",
 resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
 resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
 resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
 resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
 //Map component script
 resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
 //Report Viewer Script
 resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
 resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
 };
 reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
 {
 resourcesPath + @"\jquery-1.7.1.min.js"
 };
}
```

The data visualization components will not export without the above script configurations.

## NuGet Packages for ASP.NET Web Forms

Refer to the following steps to configure Bold Reporting NuGet packages for ASP.NET Web Forms application.

### Configure NuGet feed URL

#### Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** NuGet.org

**Source:** <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/aspnet-web-forms/report-viewer/images/nuget-packages/NuGet\_Config.png)

### Offline NuGet feed URL

Bold Reporting NuGet packages are shipped into our Bold Reporting Tools build. To configure the packages from Bold Reports installed location, use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** Bold Reports installed NuGet

**Source:** {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages

![Offline NuGet Configure](/static/assets/aspnet-web-forms/report-viewer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

## Installing NuGet packages

### Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**.

Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages (**BoldReports.WebForms** or **BoldReports.Web**), and then click **Install** button.

### Install using Package Manager Console

To install the Bold Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

### install specified package in default project

```
Install-Package <Package Name>
```

### install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

### install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

```
'
```

#### For example:

```
'cmd
```

### install specified package in default project

```
Install-Package BoldReports.WebForms
```

### install specified package in default project with specified Package Source

```
Install-Package BoldReports.Web -Source "https://api.nuget.org/v3/index.json"
```

### install specified package in specified project

```
Install-Package BoldReports.WebForms -ProjectName BoldReportsApplication
```

```
'
```

## Upgrading NuGet packages

### Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**

Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

### Upgrading using Package Manager Console

To update the installed Bold Reporting NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

### Update specific NuGet package in default project

```
Update-Package <Package Name>
```

### Update all the packages in default project

```
Update-Package
```

### Update specified package in default project with specified package source

```
Update-Package <Package Name> -Source <Source Location>
```

### Update specified package in specified project

```
Update-Package <Package Name> - ProjectName <Project Name>
```

,

#### For example:

```
'cmd
```

### Update specified Bold Reporting NuGet package

```
Update-Package BoldReports.Web
```

## Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

## Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportsApplication

### Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing `nuget.exe` to latest version use the following command:

```
`cmd
```

`nuget update -self`

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting NuGet packages.

```
`cmd
```

### update all NuGet packages from config file

`nuget update <configPath> [options]`

### update all NuGet packages from specified Packages Source

`nuget update -Source <Source Location> [optional]`

`configPath` is optional. It identifies the `package.config` or solutions file lists the packages utilized in the project.

**For example:**

```
`cmd
```

### Update all NuGet packages from config file

`nuget update "C:\Users\BoldReportsApplication\package.config"`

### Update all NuGet packages from specified Packages Source

`nuget update -Source "https://api.nuget.org/v3/index.json"`

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

## Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports ASP.NET Webforms Report Viewer.

[How can improve the performance and handle the large amounts of data with Report Viewer?](#)

[Is Report Viewer compatible with latest version of JQuery library?](#)

[Is the back action from drillthrough report will load the report again with Report Viewer?](#)

[Is possible to change the culture of the date time parameter](#)

[Is it possible to hide report parameters?](#)

[Is it possible to hide the export options in Report Viewer?](#)

[Is it possible to load reports providing parameter values at runtime?](#)

### Is possible to change the culture of the date time parameter

Yes, we can change the culture of the date time parameter for Report Viewer by changing the locale. You can make use the following reference to change the locale of Report Viewer.

[ReportViewer Localization](#)

In Report Viewer, we could not change the culture of specific parameter or UI. So, we have to achieve the requirements only by changing the culture.

### Is it possible to hide the parameters in Report Viewer

Yes, it is possible to hide the parameters in Report Viewer. On hiding the parameters, the users will not be able to have the parameter block in the Report Viewer. Refer to this [Hide parameter block and toolbar items](#) section.

### Is it possible to hide the export options in Report Viewer

Yes, it is possible to hide the export options in Report Viewer. The Report Viewer provides the `exportOptions` property to show or hide the default export types available in the component. Refer to this [Decide or Hide the export options](#) section.

### Is it possible to load reports providing parameter values at runtime

Yes, it is possible to load reports with application inputs as parameters in Report Viewer. You need to provide the input values to the Report Viewer from client side at runtime using the `parameters` property, which accepts JSON array values. Refer to this [Set parameter at client](#) section.

## How to Create RDL/RDLC Report

The following sections explain about how to create a new RDL/RDLC report using Bold Report Designer, Microsoft Report Builder and Visual Studio Report Server project template.

[Create a RDL report](#)

[Create a RDLC report](#)

[Change the exporting document file name based on the parameter](#)

[Change the connection string datasource dynamically](#)

[Disable the vertical scrollbar in parameter panel](#)

## Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

Bold Reports Report Designer.

Microsoft Report Builder.

Visual Studio Report Server project template.

## Bold Reports Report Designer

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

## Microsoft SQL Report Builder

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

## Visual Studio Report Server template

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

## Create a RDLC report using business object data source

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

### Prerequisites

Microsoft Visual Studio 2017 or higher

### [Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

#### Create business object class

Open Visual Studio from the File menu and select **New Project**.

Create project with class library type from the project type list.

![Add a new class library project](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/class-library-project.png)

Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
 public string ProdCat { get; set; }
 public string SubCat { get; set; }
 public string OrderYear { get; set; }
 public string OrderQtr { get; set; }
 public double Sales { get; set; }
}'
```

Clean and build the application.

#### Add an RDLC report

Right-click the project and click **Add > New Item**.

Search Report with new item and select **Report Wizard** to start the creation with dataset selection.

Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

#### Data source and table configuration wizard

Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/choose-data-source-type.png)

Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/select-data-objects.png)

In the **DataSet Properties** wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/arrange-table-fields.png)

Choose the table layout and click **Next**.

Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/aspnet-web-forms/report-viewer/images/how-to/create-report/sales-report-design.png)

## Adding data visualization scripts

To render the report with data visualization components such as chart, gauge and map items, must add scripts of the visualization element. The following table shows the script reference that need to be added in Report Viewer page for data visualization elements.

Visualization Item | Script File

Chart | ej.chart.min.js

Gauge | ej2-base.min.js, ej2-data.min.js, ej2-pdf-export.min.js, ej2-svg-base.min.js, ej2-lineargauge.min.js and ej2-circulargauge.min.js

Map | ej.map.min.js

To render the chart report item, add chart control script **ej.chart.min.js** before the **bold.report-viewer.min.js** reference in **Default.aspx** page as in following code sample.

```
'html
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
```

|                                                                                        |                       |
|----------------------------------------------------------------------------------------|-----------------------|
| How to change the exporting document file name based on parameter configuration wizard | Data source and table |
|----------------------------------------------------------------------------------------|-----------------------|

```
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

The following code can be used to render the chart, gauge and map report items in Report Viewer.

```
'html
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<script src="https://code.jquery.com/jquery-1.10.2.min.js" type="text/javascript"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<!--Used to render the chart item. Add this script, only if your report contains the chart report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!--Used to render the gauge item. Add this script, only if your report contains the gauge report item. -->
<script src="Scripts/bold-reports/common/ej2-base.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-data.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-pdf-export.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-svg-base.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js"></script>
<!--Used to render the map item. Add this script, only if your report contains the map report item.-->
<script src="Scripts/bold-reports/data-visualization/ej.map.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
`
```

## How to change the exporting document file name based on parameter

Find the following steps to change the file based on parameter values in Report.

Create the file exporting file name using the parameters in **onRenderingComplete** event and store it in local variable.

```
'html
```

```
function onRenderingComplete(event) {
 var parameters = event.reportParameters;
 if(parameters){
 for (var i = 0; i < parameters.length; i++) {
 if(parameters[i].Name == "Department"){
 this.exportFileName = "Sales for " + parameters[i].Value;
 }
 }
 }
}
```

Use the file Name property with export, click event to change the file using the value stored in local variable used for having the file using the parameters.

```
'html
function onExportItemClick(event) {
 event.fileName = this.exportFileName ;
}
'
```

## How to change the connection string datasource dynamically

Find the following steps to change connection string in datasource.

You need a report action information to get the data source information from report. So, stored the JsonResult information to local property in **PostReportAction** method as shown in the following code example.

```
'csharp
private Dictionary<string, object> _jsonResult;
//Post action for processing the rdl/rdlc report
public object PostReportAction(Dictionary < string, object > jsonResult)
{
 if (jsonResult != null && jsonResult.Keys.Count > 0)
 {
 _jsonResult = jsonResult;
 }
 return ReportHelper.ProcessReport(jsonResult, this);
}
```

Use the JsonResult information with **ReportHelper.GetDatasource** API to get the data source details of the reports and you have to use the **DataSourceCredentials** to change the connection string of the data source.

```
'csharp
public void OnReportLoaded(ReportViewerOptions reportOption)
{
if (jsonResult != null && jsonResult.Keys.Count > 0)
{
List<DataSourceInfo> datasources = ReportHelper.GetDataSources(_jsonResult, this);
foreach (DataSourceInfo item in datasources)
{
if (item.DataProvider == "SQL")
{
string connectionString = "Data Source = dataplatformdemodata.syncfusion.com; Initial Catalog =
AdventureWorks; User ID = 'demoreadonly@data-platform-demo'; Password = 'N@c=Y8s*1&dh'";
DataSourceCredentials DataSourceCredentials = new DataSourceCredentials();
DataSourceCredentials.Name = item.DataSourceName;
DataSourceCredentials.UserId = null;
DataSourceCredentials.Password = null;
DataSourceCredentials.ConnectionString = connectionString;
DataSourceCredentials.IntegratedSecurity = false;//if windows credentials means we need to pass true
as IntegratedSecurity
reportOption.ReportModel.DataSourceCredentials = new List<DataSourceCredentials>
{
DataSourceCredentials
};
}
}
}
}
`
```

## How to disable the vertical scrollbar in parameter panel

To disable the vertical scrollbar in parameter panel, set the `enableparameterblockscroller` property to false.

```
Example
`js
<div id="report viewer"></div>
<script>
$("#report viewer").boldReportViewer(
{
enableParameterBlockScroller: false
});
</script>
`
```

## Bold Report Writer

Report Writer is a class library that enables the user to render reports defined in Microsoft's RDL format (2008 or 2008 R2) as PDF, Word, Excel, HTML and CSV documents.

The important features of Report Writer are listed as follows:

RDL Specification - Supports RDL specification for the SQL Server 2008 and RDL specification for the SQL Server 2008 R2 only. List of available report definition formats:  
[msdn.microsoft.com/library/dd297486\(SQL.100\).](http://msdn.microsoft.com/library/dd297486(SQL.100).)

Data sources - You can use advanced database servers Data Sources in Report Writer (SQL and Oracle).

Charts - Show all basic types of charts that are available in Microsoft RDL reports.

Tablix - Shows the summaries and simple tables.

Gauge - Shows measurement values by using expression values.

Textbox - Shows textbox data with expression support.

Export - Export report as PDF, Word, Excel, HTML and CSV.

Report Parameter - Views the report based on the report parameter value.

## Export SSRS RDL Report in Bold Reports ASP.NET Report Writer

The Report Writer is a class library that is used to export the RDL report with popular file formats like PDF, Microsoft Word, Microsoft CSV, and Microsoft Excel without previewing the report in webpage. This section describes how to export the RDL report in ASP.NET application using the [Report Writer](#).

### Create an ASP.NET application

Start the Visual Studio 2019 and click **Create new project**.

Choose **ASP.NET Web Application (.NET Framework)**, and then click **Next**.

Change the project name, and then click **Create**.

Choose **Web Forms** and **Web API** then click **OK**.

![Creating a new ASP.NET Application Project](/static/assets/aspnet-web-forms/report-writer/images/getting-started/aspnet-web-forms-application-template.png)

### List of dependency libraries

In the Solution Explorer tab right-click the project or solution , and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **BoldReports.Webpackage**, and install this in your application.

The following table provides details about the dependency packages and their usage.

#### Package | Purpose

**Syncfusion.Pdf.AspNet** | Exports the report to a PDF.

**Syncfusion.DocIO.AspNet** | Exports the report to a Word.

**Syncfusion.XlsIO.AspNet** | Exports the report to an Excel.

**Syncfusion.Presentation.AspNet** | Exports the report to a PowerPoint.

**Newtonsoft.Json** | Serializes and deserialize data for the Report Writer. It is a mandatory package for Report Writer, and the package version should be 10.0.1 or higher.

In this tutorial, the **sales-order-detail.rdl** report is used and it can be downloaded [here](#). You can get the reports from Bold Reports installation location. For more information, refer to [samples and demos](#) section.

### Server side Report Writer changes

Create a folder **Resources** in your application. Copy and paste the sample RDL reports into the **Resources** folder.

Open the **Default.aspx.cs** and add the following using namespace .

```
'csharp
using BoldReports.Writer;
'
```

Initialize the Report Writer using the following code example.

```
'csharp
protected void ExportButton_Click(object sender, EventArgs e)
{
 string fileName = null;
 WriterFormat format;
 HttpContext httpContext = System.Web.HttpContext.Current;
 BoldReports.Writer.ReportWriter writer = new BoldReports.Writer.ReportWriter();
 writer.ReportPath= Server.MapPath("~/Resources/sales-order-detail.rdl");
 if (this.ExportFormat.SelectedValue == "PDF")
 {
 fileName = "sales-order-detail.pdf";
 format = WriterFormat.PDF;
 }
 else if (this.ExportFormat.SelectedValue == "Word")
 {
 fileName = "sales-order-detail.doc";
 format = WriterFormat.Word;
 }
 else if (this.ExportFormat.SelectedValue == "Html")
 {
 fileName = "sales-order-detail.Html";
 format = WriterFormat.HTML;
 }
 else if (this.ExportFormat.SelectedValue == "PPT")
 {
 fileName = "sales-order-detail.ppt";
 format = WriterFormat.PPT;
 }
}
```

```
else
{
 fileName = "sales-order-detail.xls";
 format = WriterFormat.Excel;
}
writer.Save(fileName, format, httpContext.Response);
}
`
```

## Client side changes

Open the `Default.aspx` home page. Replace the following code snippet in `Default.aspx` home page.  
Ensure that `inherits` value is same for application name `Inherits=".Default"`.

```
`html
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="WebformsWriter._Default" %>
<!DOCTYPE html>
<html>
<head runat="server">
<title> BoldReports Writer </title>
</head>
<body>
<form id="form1" runat="server">
<div>
</div>
</form>
</body>
</html>
`
```

Add the following code example in the `tag` of the `Default.aspx` page to view Report Writer export options.

```
`html
<body>
<div class="container">
```

```
<div class="content_section">
<form id="form1" runat="server">
<div id="description_Pane" style="text-align: justify;">
<h3>Description</h3>

Bold ReportWriter is a powerful control for exporting RDL and RDLC files into specified
format files.

</div>
<div id="export_Pane" style="margin-top: 4%; ">
<h3>Export Report</h3>

Choose a file format to view the selected document generated from Report file by using Essential
ReportWriter.

<div id="selection_Pane" style="margin-top: 2%; ">
<asp:Label Style="font-size: large;" runat="server">
Save As :
</asp:Label>
<asp:RadioButtonList RepeatLayout="Flow" ID="ExportFormat" RepeatDirection="Horizontal"
runat="server">
<asp:ListItem Selected="True">PDF</asp:ListItem>
<asp:ListItem>Word</asp:ListItem>
<asp:ListItem>Excel</asp:ListItem>
<asp:ListItem>Html</asp:ListItem>
<asp:ListItem>PPT</asp:ListItem>
<asp:ListItem>csv</asp:ListItem>
</asp:RadioButtonList>
<asp:Button style="width: 18%; margin-left: 2%;" ID="ExportButton" runat="server"
OnClick="ExportButton_Click" Text="Generate" />
</div>
</div>
</form>
</div>
```

```
</div>
</body>
`
```

Now, Run and export the report with specified export format in your Report Writer application.

Congratulations! you have completed your first WEB Forms Writer application!.Click [here](#) to download the already created WEB Forms Report Writer application.

## Reporting tools for ASP.NET Web Forms

The **Report Designer** is a web based reporting tool to create and edit the RDL(Report Definition Language) standard reports. It comes with a wide range of report items to transform data into meaningful information and quickly build the reports with the help of following features.

### Key features

**Data Sources** --- Supports connection to major data providers such as

**Microsoft SQL Server, Oracle, OLEDB** and **ODBC** for exploring data and design reports with a wide range of data sources.

**User friendly Environment** --- Provides an effective design area, configuration options, and drag-and-drop facilities to make it easy for business users to compose reports.

**Report items** --- All interactive report items that are commonly used in business reports is built-in, including charts, tablix, list, subreports, textboxes, images, lines, and rectangles for better visual representation of data.

**Report Parameter** --- Supports parameter to specify the data to filter in a report, connect related reports together and vary report presentation.

**Expression** --- Expressions are used throughout the report definition in parameters, queries, filters and report item properties to perform additional operations such as mathematical computation, conditional formatting, inspection, conversions, and more.

## Add Web Report Designer to an ASP.NET WebForms application

This section explains the steps required to add Web Report Designer to an ASP.NET WebForms application.

### Create ASP.NET Web Forms application

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![Project Creation Wizard](/static/assets/aspnet-web-forms/report-designer/images/Getting-Started\_img1.png)

Then choose the **Web Forms** in template, enable **Web API** option in the **Add folders and core references for:** section.

![Razor view engine](/static/assets/aspnet-web-forms/report-designer/images/Getting-Started\_img3.png)

## Add Assembly References

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Designer NuGet packages.

Search for **BoldReports.Web** and **BoldReports.WebForms** NuGet packages, and install them in your Web Forms application.

### Package | Purpose

GetResource | Returns the report resource for the requested key.

ProcessReport | Processes the report request and returns the result.

Please add the following code example in **ReportingAPIController.cs**.

```
'csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Web.Http;
using BoldReports.Web;
using BoldReports.Web.ReportViewer;
using BoldReports.Web.ReportDesigner;
using System.IO;
using System.Reflection;
using System.Web;
namespace ReportDesignerSample.Controllers
{
 public class ReportingAPIController : ApiController, IReportDesignerController
 {
```

```
private string GetFilePath(string itemName, string key)
{
 string targetFolder = HttpContext.Current.Server.MapPath("~/");
 targetFolder += "Cache";
 if (!Directory.Exists(targetFolder))
 {
 Directory.CreateDirectory(targetFolder);
 }
 if (!Directory.Exists(targetFolder + "\\\" + key))
 {
 Directory.CreateDirectory(targetFolder + "\\\" + key);
 }
 return targetFolder + "\\\" + key + "\\\" + itemName;
}

[System.Web.Http.ActionName("GetImage")]
[AcceptVerbs("GET")]
public object GetImage(string key, string image)
{
 return ReportDesignerHelper.GetImage(key, image, this);
}

/// <summary>
/// Action (HttpPost) method for posting the request for designer actions.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing designer request.</param>
/// <returns>The object data.</returns>
public object PostDesignerAction(Dictionary<string, object> jsonData)
{
 //Processes the designer request and returns the result.
 return ReportDesignerHelper.ProcessDesigner(jsonData, this, null);
}
public bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage)
{
 errorMessage = string.Empty;
```

```
if (itemData.Data != null)
{
 File.WriteAllBytes(this.GetFilePath(itemId, key), itemData.Data);
}

else if (itemData.PostedFile != null)
{
 var fileName = itemId;
 if (string.IsNullOrEmpty(itemId))
 {
 fileName = Path.GetFileName(itemData.PostedFile.FileName);
 }
 itemData.PostedFile.SaveAs(this.GetFilePath(fileName, key));
}

return true;
}

public ResourceInfo GetData(string key, string itemId)
{
 var resource = new ResourceInfo();
 resource.Data = File.ReadAllBytes(this.GetFilePath(itemId, key));
 return resource;
}

/// <summary>
/// Action (HttpPost) method for uploaded file actions.
/// </summary>
public void UploadReportAction()
{
 //Processes the designer file upload request's.
 ReportDesignerHelper.ProcessDesigner(null, this, System.Web.HttpContext.Current.Request.Files[0]);
}

/// <summary>
/// Action (HttpGet) method for getting resource to report.
/// </summary>
/// <param name="key">The unique key to get the required resource.</param>
```

```
/// <param name="resourcetype">The type of the requested resource.</param>
/// <param name="isPrint">If set to <see langword="true"/>, then the resource is generated for printing.</param>
/// <returns>The object data.</returns>
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]
public object GetResource(string key, string resourcetype, bool isPrint)
{
 //Returns the report resource for the requested key.
 return ReportHelper.GetResource(key, resourcetype, isPrint);
}
/// <summary>
/// Report Initialization method that is triggered when report begin processed.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
 reportOption.ReportModel.ExportResources.Scripts = new List<string>
 {
 resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
 resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
 //Chart component script
 resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
 //Gauge component scripts
 resourcesPath + @"\bold-reports\common\ej2-base.min.js",
 resourcesPath + @"\bold-reports\common\ej2-data.min.js",
 resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
 resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
 resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
 resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
 //Map component script
 resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
 }
}
```

```
//Report viewer Script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
};

reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\jquery-1.7.1.min.js"
};
}

/// <summary>
/// Report loaded method that is triggered when report and sub report begins to be loaded.
/// </summary>
/// <param name="reportOptions">The ReportViewer options.</param>
public void OnReportLoaded(ReportViewerOptions reportOption)
{
//You can update report options here
}

/// <summary>
/// Action (HttpPost) method for posting the request for report process.
/// </summary>
/// <param name="jsonData">The JSON data posted for processing report.</param>
/// <returns>The object data.</returns>
public object PostReportAction(Dictionary<string, object> jsonData)
{
//Processes the report request and returns the result.
return ReportHelper.ProcessReport(jsonData, this as IReportController);
}
}
```

Add routing information

Open the `WebApiConfig.cs` file from `App_Start` folder of your application.

Modify the **routeTemplate** in **Register** event to include the **{action}** parameter in the URI as follows.

```
'csharp
public static class WebApiConfig
{
 public static void Register(HttpConfiguration config)
 {
 // Web API configuration and services
 // Web API routes
 config.MapHttpAttributeRoutes();

 config.Routes.MapHttpRoute(
 name: "DefaultApi",
 routeTemplate: "api/{controller}/{action}/{id}",
 defaults: new { id = RouteParameter.Optional }
);
 }
}
`
```

### Set the service URL

To browse, open and save the reports in the application, set the WebAPI controller name to the **ServiceUrl** property of the web Report Designer. You can use the following code in the **Default.aspx** page.

```
'js
<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">
 <link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
 <link href="Content/bold-reports/material/bold.reportdesigner.min.css" rel="stylesheet" />
 <link href="Scripts/CodeMirror/lib/codemirror.css" rel="stylesheet" />
 <link href="Scripts/CodeMirror/addon/hint/show-hint.css" rel="stylesheet" />
 <script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
 type="text/javascript"></script>
 <script src="https://cdn.boldreports.com/external/jsrender.min.js" type="text/javascript"></script>
 <script src="Scripts/CodeMirror/lib/codemirror.js"></script>
 <script src="Scripts/CodeMirror/addon/hint/show-hint.js"></script>
 <script src="Scripts/CodeMirror/addon/hint/sql-hint.js"></script>
 <script src="Scripts/CodeMirror/mode/sql/sql.js"></script>
```

```
<script src="Scripts/CodeMirror/mode/vb/vb.js"></script>
<!--Render the gauge item. Add this script only if your report contains the gauge report item. -->
<script src="Scripts/bold-reports/common/ej2-base.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-data.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-pdf-export.min.js"></script>
<script src="Scripts/bold-reports/common/ej2-svg-base.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js"></script>
<script src="Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<script src="Scripts/bold-reports/common/bold.report-designer-widgets.min.js"></script>
<!--Chart component script added before the web report designer script to render chart report item in reports-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
<!-- Report Designer component script-->
<script src="Scripts/bold-reports/bold.report-designer.min.js"></script>
<div style="height: 650px; width: 950px">
<Bold:ReportDesigner runat="server" ID="designer" ServiceUrl="/api/ReportingAPI">
</Bold:ReportDesigner>
</div>
</asp:Content>
`
```

## Run the Application

Run the sample application and you can see the ReportDesigner on the page as displayed in the following screenshot.

![Report Designer demo](/static/assets/aspnet-web-forms/report-designer/images/Getting-Started\_img9.png)

## Dependencies

The ReportDesigner have the following list of internal and external dependencies to render the component.

The BoldReports.JavaScript contains reporting components scripts and style sheets. Install the BoldReports.JavaScript package, the scripts and styles required for Report Designer will be added to Scripts, Content folders in your application.

## Scripts

<span style="font-weight:bold">Internal dependencies</span>

| Name                                | Details                                                                                                                                                     | Local file path                                                 |
|-------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| bold.reports.common.min.js          | Common script for reporting widgets.                                                                                                                        | Scripts/bold-reports/common/bold.reports.common.min.js          |
| bold.reports.widgets.min.js         | Supports Syncfusion widgets to render in HTML5 format and it contains the dependent Syncfusion controls that is common for both report designer and viewer. | Scripts/bold-reports/common/bold.reports.widgets.min.js         |
| bold.report-designer-widgets.min.js | Supports Syncfusion widgets to render in HTML5 format and it contains the dependent Syncfusion controls for report designer                                 | Scripts/bold-reports/common/bold.report-designer-widgets.min.js |
| ej.chart.min.js                     | Renders the chart item. Add this script, only if your report contains the chart report item.                                                                | Scripts/bold-reports/data-visualization/ej.chart.min.js         |
| ej2-base.min.js                     | Renders the gauge item. Add this script only if your report contains the gauge report item.                                                                 | Scripts/bold-reports/common/ej2-base.min.js                     |
| ej2-data.min.js                     | Renders the gauge item. Add this script only if your report contains the gauge report item.                                                                 | Scripts/bold-reports/common/ej2-data.min.js                     |
| ej2-pdf-export.min.js               | Renders the gauge item. Add this script only if your report contains the gauge report item.                                                                 | Scripts/bold-reports/common/ej2-pdf-export.min.js               |
| ej2-svg-base.min.js                 | Renders the gauge item. Add this script only if your report contains the gauge report item.                                                                 | Scripts/bold-reports/common/ej2-svg-base.min.js                 |
| ej2-lineargauge.min.js              | Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.                                                   | Scripts/bold-reports/data-visualization/ej2-lineargauge.min.js  |

|                             |                                                                                                               |                                                                  |
|-----------------------------|---------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| ej2-circulargauge.min.js    | Renders the circular gauge item. Add this script only if your report contains the circular gauge report item. | Scripts/bold-reports/data-visualization/ej2-circulargauge.min.js |
| ej.map.min.js               | Renders the map item. Add this script only if your report contains the map report item.                       | Scripts/bold-reports/data-visualization/ej.map.min.js            |
| bold.report-viewer.min.js   | Used to preview the reports in report designer.                                                               | Scripts/bold-reports/bold.report-viewer.min.js                   |
| bold.report-designer.min.js | Used to render the Bold Report Designer widget.                                                               | Scripts/bold-reports/bold.report-designer.min.js                 |

<span style="font-weight:bold">External dependencies</span>

| Name                            | Details                                                                                                                                                                                                  | CDN link / Local file path                                                                                                                                                                                                                                                                               |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| jQuery 1.7.1 and later versions | Common jQuery script to render the Syncfusion JavaScript Reporting widgets                                                                                                                               | Secured<br>Link: <a href="https://cdn.boldreports.com/external/jquery-1.10.2.min.js">https://cdn.boldreports.com/external/jquery-1.10.2.min.js</a><br>Unsecured<br>Link: <a href="http://cdn.boldreports.com/external/jquery-1.10.2.min.js">http://cdn.boldreports.com/external/jquery-1.10.2.min.js</a> |
| jsrender                        | The script to render the templates in the browser.                                                                                                                                                       | Secured<br>Link: <a href="https://cdn.boldreports.com/external/jsrender.min.js">https://cdn.boldreports.com/external/jsrender.min.js</a><br>Unsecured<br>Link: <a href="http://cdn.boldreports.com/external/jsrender.min.js">http://cdn.boldreports.com/external/jsrender.min.js</a>                     |
| Codemirror                      | Report Designer requires the code mirror scripts to edit the SQL queries and Visual Basic code functions with syntax highlighter.<br><br>codemirror.js<br>show-hint.js<br>sql-hint.js<br>sql.js<br>vb.js | Scripts/CodeMirror/lib/codemirror.js<br>Scripts/CodeMirror/addon/hint/show-hint.js<br>Scripts/CodeMirror/addon/hint/sql-hint.js<br>Scripts/CodeMirror/mode/sql/sql.js<br>Scripts/CodeMirror/mode/vb/vb.js                                                                                                |

## Styles

<span style="font-weight:bold">Internal dependencies</span>

| Name                        | Details                                                                 | Local file path                                               |
|-----------------------------|-------------------------------------------------------------------------|---------------------------------------------------------------|
| bold.reports.all.min.css    | It contains the styles and css references of dependent components.      | Content/bold-reports/[theme-name]/bold.reports.all.min.css    |
| bold.reportdesigner.min.css | It contains the styles and css references of Report Designer component. | Content/bold-reports/[theme-name]/bold.reportdesigner.min.css |

Refer the following syntax:

Content/bold-reports/[theme-name]/[fileName]

<span style="font-weight:bold">Example</span>

| Theme Name    | Format                                                                                                              |
|---------------|---------------------------------------------------------------------------------------------------------------------|
| Default-Theme | Content/bold-reports/material/bold.reports.all.min.css<br>Content/bold-reports/material/bold.reportdesigner.min.css |

<span style="font-weight:bold">External dependencies</span>

| Name       | Details                                                                                                                                                                     | CDN link                                                                             |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| Codemirror | Report Designer requires the code mirror styles to edit the SQL queries and Visual Basic code functions with syntax highlighter.<br><br>codemirror.css<br><br>show-hint.css | Scripts/CodeMirror/lib/codemirror.css<br>Scripts/CodeMirror/addon/hint/show-hint.css |

## Service side dependencies

### Internal dependencies

Package | Purpose

Syncfusion.DocIO.Base | Exports the report to a Word.

Syncfusion.Pdf.Base | Exports the report to a PDF.

Syncfusion.XlsIO.Base | Exports the report to an Excel.

Syncfusion.Presentation.Base | Exports the report to an PPT.

## External dependencies

### Package | Purpose

**Newtonsoft.Json** | Serializes and deserialize data for the Report Designer. It is a mandatory package for Report Designer, and the package version should be higher than 10.0.1 for NET Core 2.0 and others should be higher than 9.0.1.

## Localization

Localization is the process of customizing the application for a given culture and locale - involving much more than the simple translation of text. In web report designer, localized strings appropriate to each culture are stored in separate files and loaded according to the UI culture setting.

By default report designer display strings in US English(en-US).

To render the static text with specific culture, refer the following corresponding culture script files and set culture name to the **locale** property of the Report Designer.

```

 ej.localetexts.fr-FR.min.js
 ej.culture.fr-FR.min.js

```

Install **BoldReports.Global** Nuget package in your ASP.NET application.

Refer to the **ej.localetexts.fr-FR.min.js** script file from the **Scripts** folder of your application.

```
'html
<script src="Scripts/bold-reports/I10n/reportdesigner/ej.localetexts.fr-FR.min.js"></script>
'
```

Refer to the **ej.culture.fr-FR.min.js** script file from the **Scripts** folder of your application.

```
'html
<script src="Scripts/bold-reports/i18n/ej.culture.fr-FR.min.js"></script>
'
```

To render the localization Report Designer, use the following code snippet.

The following code example illustrates how to set the **locale** property for Report Designer in the index page.

```
'html
<div style="height: 650px; width: 950px">
<Bold:ReportDesigner runat="server" ID="designer" ServiceUrl="/api/ReportingAPI" Locale="fr-FR">
```

```
</Bold:ReportDesigner>
</div>
`
```

The following code example illustrates how to add Report Designer localization scripts in the Default.aspx page.

```
'html
<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">
<link href="Content/bold-reports/material/bold.reports.all.min.css" rel="stylesheet" />
<link href="Content/bold-reports/material/bold.reportdesigner.min.css" rel="stylesheet" />
<link href="Scripts/CodeMirror/lib/codemirror.css" rel="stylesheet" />
<link href="Scripts/CodeMirror/addon/hint/show-hint.css" rel="stylesheet" />
<script src="https://cdn.boldreports.com/external/jquery-1.10.2.min.js"
type="text/javascript"></script>
<script src="https://cdn.boldreports.com/external/jsrender.min.js" type="text/javascript"></script>
<script src="Scripts/CodeMirror/lib/codemirror.js"></script>
<script src="Scripts/CodeMirror/addon/hint/show-hint.js"></script>
<script src="Scripts/CodeMirror/addon/hint/sql-hint.js"></script>
<script src="Scripts/CodeMirror/mode/sql/sql.js"></script>
<script src="Scripts/CodeMirror/mode/vb/vb.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.common.min.js"></script>
<script src="Scripts/bold-reports/common/bold.reports.widgets.min.js"></script>
<script src="Scripts/bold-reports/common/bold.report-designer-widgets.min.js"></script>
<!--Chart component script added before the web report designer script to render chart report item in
reports-->
<script src="Scripts/bold-reports/data-visualization/ej.chart.min.js"></script>
<!-- Report Viewer component script-->
<script src="Scripts/bold-reports/bold.report-viewer.min.js"></script>
<!-- Report Designer component script-->
<script src="Scripts/bold-reports/bold.report-designer.min.js"></script>
<script src="Scripts/bold-reports/l10n/reportdesigner/ej.localetexts.fr-FR.min.js"></script>
<script src="Scripts/bold-reports/i18n/ej.culture.fr-FR.min.js"></script>
<div style="height: 650px; width: 950px">
<Bold:ReportDesigner runat="server" ID="designer" ServiceUrl="/api/ReportingAPI" Locale="fr-FR">
```

```
</Bold:ReportDesigner>
</div>
</asp:Content>
`
```

## Integrate the component with Report Server

You can integrate Report Designer with Report Server to create, edit, browse and publish reports using the Report Server built-in API service.

The Report Designer requires the `serviceAuthorizationToken` to perform the API actions with Bold Report Server. Create a token for the user by using the server RESTful API, you can use the following code in `Default.aspx.cs` file to generate the token.

```
`csharp
public partial class _Default : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
{
this.designer.ServiceAuthorizationToken = this.GenerateToken("guest@boldreports.com", "demo");
}
public string GenerateToken(string userName, string password)
{
using (var client = new HttpClient())
{
client.DefaultRequestHeaders.Accept.Clear();
var content = new FormUrlEncodedContent(new[]
{
new KeyValuePair<string, string>("grant_type", "password"),
new KeyValuePair<string, string>("username", userName),
new KeyValuePair<string, string>("password", password)
});
var result = client.PostAsync("https://on-premise-demo.boldreports.com/reporting/api/site/site1/token", content).Result;
string resultContent = result.Content.ReadAsStringAsync().Result;
var token = JsonConvert.DeserializeObject<Token>(resultContent);
return token.tokenType + " " + token.accessToken;
}
```

```
}

}

}

public class Token
{
 public string access_token { get; set; }

 public string token_type { get; set; }

 public string expires_in { get; set; }

 public string userName { get; set; }

 public string serverUrl { get; set; }

 public string password { get; set; }
}
```

Set the Bold Report Server built-in service URL to the `serviceUrl` property and assign the created token to `serviceAuthorizationToken` property.

```
'js
<div style="height: 650px; width: 950px">
<Bold:ReportDesigner runat="server" ID="designer" ServiceUrl="https://on-premise-
demo.boldreports.com/reporting/reportservice/api/Designer"
OnClientAjaxBeforeLoad="onAjaxRequest">
</Bold:ReportDesigner>
</div>
<script type="text/javascript">
function onAjaxRequest(args) {
 args.headers.push({ Key: 'serverurl', Value: 'https://on-premise-
demo.boldreports.com/reporting/api/site/site1' });
}
</script>
'
```

## Designing Reports

The Bold Report Designer provides an end-user documentation that describes how to interact with the Report Designer's UI and design an interactive reports. Refer the following user guide sections to get start with Bold Report Designer.

## Connecting your data

A report must have a data source and dataset to include data. Each data source contains data connection information. Each dataset contains a query and collection of fields to represent the data returned from the data source.

[Manage data source](#)

[Manage data set](#)

## Transform your data

You can transform the data as required after it is retrieved from data source with following features:

[Join tables](#)

[Formatting columns](#)

[Query filter](#)

[Data set parameter](#)

[Query parameter](#)

[Configure expression columns](#)

## Working with report parameters

Supports creating report parameters manually or based on a data set query. Parameters are used to interactively provide user inputs at run-time to vary report presentation based on it.

[Add report parameter](#)

[Edit report parameter](#)

[Delete report parameter](#)

[Cascading parameter](#)

[Multi-value parameter](#)

## Embed images into the report

Supports to embed local or database images into the report and image data is stored within the report definition. The embedded images in a report are listed in the **Image Manager panel**.

[Embed images into the report](#)

## Interacting with report design surface

WYSIWYG user interface that allows report to be edited in a form that resembles its appearance when printed or displayed.

[Interacting with report design surface](#)

## Report items

Offers rich set of report items to present the data in comprehensive reports to help companies make better business decisions.

[Configure report items](#)

## Customize appearance

The Properties panel shows all the properties of the selected section, report items or the report itself to customize the appearance of the report.

[Customize appearance](#)[Report design settings](#)[Configure design settings](#)[Shape report data](#)[Filter data](#)[Group data](#)[Sort data](#)[Format data](#)[Interactive features](#)[Hyperlink](#)[Drilldown](#)[Interactive sorting](#)[Working with Expressions](#)

Expressions are used to specify criteria for retrieving and formatting data, creating calculated fields and calculating summaries, conditionally shaping data and changing a report control's appearance.

[Expressions builder](#)[Open report](#)[Open report](#)[Save report](#)[Save report](#)[Preview report](#)[Preview report](#)

## Migrate Report Designer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Designer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Designer from Syncfusion Essential Studio release version to Bold Reports version of ASP.NET Web Forms Report Designer application.

### Scripts

| Old Scripts       | New Scripts                                                                                                                                                   | Description                                                                                                                                                                                                        |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ej.web.all.min.js | bold.reports.common.min.js<br>bold.reports.widgets.min.js<br>bold.report-designer-widgets.min.js<br>ej.chart.min.js<br>ej2-base.min.js<br><br>ej2-data.min.js | We have removed the dependency scripts for report designer from existing ej.web.all.min.js script and provided the dependency scripts as below:<br><br>bold.reports.common.min.js<br>- Common script for reporting |

|  |                                                                                                                                                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <p>ej2-pdf-export.min.js<br/>ej2-svg-base.min.js<br/><br/>ej2-lineargauge.min.js<br/>ej2-circulargauge.min.js<br/>ej.map.min.js<br/>bold.report-viewer.min.js</p> | <p>widgets.<br/><br/>bold.reports.widgets.min.js - It contains the scripts of dependent controls that are common for both Report Designer and Report Viewer.<br/><br/>bold.report-designer-widgets.min.js - It contains the scripts of Report Designer dependent controls.<br/><br/>ej.chart.min.js - Renders the chart item. Add this script only if your report contains the chart report item.<br/><br/>ej2-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.<br/><br/>ej2-data.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.<br/><br/>ej2-pdf-export.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.<br/><br/>ej2-svg-base.min.js - Renders the gauge item. Add this script only if your report contains the gauge report item.<br/><br/>ej2-lineargauge.min.js - Renders the linear gauge item. Add this script only if your report contains the linear gauge report item.<br/><br/>ej2-circulargauge.min.js - Renders the circular gauge item. Add this script only if your report contains the circular gauge report</p> |
|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                       |                                          |                                                                                                                                                                                                                                              |
|---------------------------------------|------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                       |                                          | <p>item.</p> <p><code>ej.map.min.js</code> - Renders the map item. Add this script only if your report contains the map report item.</p> <p><code>bold.report-viewer.min.js</code> - Previews the reports designed with Report Designer.</p> |
| <code>ej.reportdesigner.min.js</code> | <code>bold.report-designer.min.js</code> | Renamed the report designer script file and it used to render the Bold Report Designer widget.                                                                                                                                               |

## Styles

| Old Scripts                     | New Scripts                           | Description                                                                                                                                                                                                  |
|---------------------------------|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ej.web.all.min.css</code> | <code>bold.reports.all.min.css</code> | We have removed the dependent controls styles for report designer from existing <code>ej.web.all.min.css</code> script and provided the dependent controls styles as <code>bold.reports.all.min.css</code> . |

Remove the old scripts and styles references from existing application, then add the new scripts and styles references based on above script name changes.

## Assemblies

| Purpose                             | Old Assemblies                                                                               | New Assemblies                        | Description                                                                                                                                                           |
|-------------------------------------|----------------------------------------------------------------------------------------------|---------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Base Assemblies                     | <code>Syncfusion.EJ.ReportViewer.dll</code><br><code>Syncfusion.EJ.ReportDesigner.dll</code> | <code>BoldReports.Web.dll</code>      | The <code>Syncfusion.EJ.ReportViewer.dll</code> and <code>Syncfusion.EJ.ReportDesigner.dll</code> assemblies have been combined as <code>BoldReports.Web.dll</code> . |
| ASP.NET Web Forms Helper Assemblies | <code>Syncfusion.EJ.dll</code><br><code>Syncfusion.EJ.Web.dll</code>                         | <code>BoldReports.WebForms.dll</code> | The <code>Syncfusion.EJ.dll</code> and <code>Syncfusion.EJ.Web.dll</code> assemblies have been combined as <code>BoldReports.WebForms.dll</code> .                    |

## Packages

| Purpose            | Old Packages                                                       | New Packages         |
|--------------------|--------------------------------------------------------------------|----------------------|
| Server Side Helper | Syncfusion.ReportDesigner.AspNet<br>Syncfusion.ReportViewer.AspNet | BoldReports.Web      |
| Control            | Syncfusion.AspNet                                                  | BoldReports.WebForms |

## Register tag prefix in Web.config file

The ASP.NET webforms reporting components tag prefix has been changed from ej to SF. Open the Web.config file and add the BoldReports.WebForms assembly reference to the <system.web.pages.controls> element with SF tag prefix.

| Old code snippet                                                                                                                                                                                                                                                                          | New code snippet                                                                                                                                                                                                                                                                                              |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>&lt;configuration&gt; ..... &lt;system.web&gt; ..... &lt;pages&gt; &lt;controls&gt; ..... ..... &lt;add assembly="Syncfusion.EJ.Web" namespace="Syncfusion.EJ.Web" tagPrefix="EJ"/&gt; &lt;/controls&gt; &lt;/pages&gt; &lt;/system.web&gt; ..... ..... &lt;/configuration&gt;</pre> | <pre>&lt;configuration&gt; ..... ..... &lt;system.web&gt; ..... ..... &lt;pages&gt; &lt;controls&gt; ..... ..... &lt;add assembly="BoldReports.WebForms" namespace="BoldReports.WebForms" tagPrefix="Bold"/&gt; &lt;/controls&gt; &lt;/pages&gt; &lt;/system.web&gt; ..... ..... &lt;/configuration&gt;</pre> |

## Namespace changes

| Assembly Name       | Old Namespace                     | New Namespace                  |
|---------------------|-----------------------------------|--------------------------------|
| BoldReports.Web.dll | Syncfusion.Reports.EJ             | BoldReports.Web                |
|                     | Syncfusion.EJ.ReportWriter        | BoldReports.Writer             |
|                     | Syncfusion.EJ.ReportViewer        | BoldReports.Web.ReportViewer   |
|                     | Syncfusion.EJ.ReportDesigner      | BoldReports.Web.ReportDesigner |
|                     | Syncfusion.Reports.EJ.Data        | BoldReports.Data               |
|                     | Syncfusion.Reporting              | BoldReports.Configuration      |
|                     | Syncfusion.EJ.RDL.ServerProcessor | BoldReports.ServerProcessor    |

Based on above assembly and namespace changes, modify the Report Designer **Web API Controller** in your application.

#### Control initialization

| Old code snippet                                                      | New code snippet                                                         |
|-----------------------------------------------------------------------|--------------------------------------------------------------------------|
| <ej:ReportDesigner runat="server" ID="designer"> </ej:ReportDesigner> | <Bold:ReportDesigner runat="server" ID="designer"></Bold:ReportDesigner> |

#### Report export configuration

The **BoldReports.Web** assembly can export the reports with data visualization components such as chart, gauge, and map, only if we configure the web scripts in Report Designer Web API controller. To configure the scripts in Web API controller, refer to the following steps:

Open the Report Designer Web API controller.

Configure the following scripts and styles in **OnInitReportOptions** on Web API controller:

**jquery-1.7.1.min.js**

**bold.reports.common.min.js**

**bold.reports.widgets.min.js**

**ej.chart.min.js** - Exports the chart item. Add this script only if your report contains the chart report item.

**ej2-base.min.js**, **ej2-data.min.js**, **ej2-pdf-export.min.js**, **ej2-svg-base.min.js**, **ej2-lineargauge.min.js** and **ej2-circulargauge.min.js** - Exports the gauge item. Add this script only if your report contains the gauge report item.

**ej.map.min.js** - Exports the map item. Add this script only if your report contains the map report item.

**bold.report-viewer.min.js**

You can replace the **OnInitReportOptions** action in Report Designer Web API controller using below code snippet.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
 reportOption.ReportModel.ExportResources.Scripts = new List<string>
```

```
{
resourcesPath + @"\bold-reports\common\bold.reports.common.min.js",
resourcesPath + @"\bold-reports\common\bold.reports.widgets.min.js",
//Chart component script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
//Gauge component scripts
resourcesPath + @"\bold-reports\common\ej2-base.min.js",
resourcesPath + @"\bold-reports\common\ej2-data.min.js",
resourcesPath + @"\bold-reports\common\ej2-pdf-export.min.js",
resourcesPath + @"\bold-reports\common\ej2-svg-base.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-lineargauge.min.js",
resourcesPath + @"\bold-reports\data-visualization\ej2-circulargauge.min.js",
//Map component script
resourcesPath + @"\bold-reports\data-visualization\ej.map.min.js",
//Report Designer Script
resourcesPath + @"\bold-reports\data-visualization\ej.chart.min.js",
resourcesPath + @"\bold-reports\bold.report-viewer.min.js"
};
reportOption.ReportModel.ExportResources.DependentScripts = new List<string>
{
resourcesPath + @"\jquery-1.7.1.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

## NuGet Packages for ASP.NET Web Forms

Refer to the following steps to configure Bold Reporting tools NuGet packages for ASP.NET Web Forms application.

[Configure NuGet feed URL](#)

[Online NuGet feed URL](#)

The Bold Reporting tools NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** NuGet.org

**Source:** <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/aspnet-web-forms/report-designer/images/nuget-packages/NuGet\_Config.png)

#### Offline NuGet feed URL

Bold Reporting tools NuGet packages are shipped into our Report Platform SDK build. To configure the packages from Bold Reports installed location, use the following steps:

Open your Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager**, and then select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** Bold Reports installed NuGet

**Source:** {System Drive}:\Program Files (x86)\Bold Reports\Reporting Tools\Nuget Packages

![Offline NuGet Configure](/static/assets/aspnet-web-forms/report-designer/images/nuget-packages/LocalNuGetConfig.png)

The system drive varies based on the installed location in your machine.

#### Installing NuGet packages

##### Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**.

Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the packages (**BoldReports.WebForms** or **BoldReports.Web**), and then click **Install** button.

install specified package in default project

Upgrading NuGet packages

#### Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

`cmd

#### install specified package in default project

Install-Package <Package Name>

#### install specified package in default project with specified package source

Install-Package <Package Name> -Source <Source Location>

#### install specified package in specified project

Install-Package <Package Name> - ProjectName <Project Name>

#### For example:

`cmd

#### install specified package in default project

Install-Package BoldReports.WebForms

#### install specified package in default project with specified Package Source

Install-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

#### install specified package in specified project

Install-Package BoldReports.WebForms -ProjectName BoldReportingApplication

#### Upgrading NuGet packages

##### Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**

Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

### Upgrading using Package Manager Console

To update the installed Bold Reporting tools NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

### Update specific NuGet package in default project

Update-Package <Package Name>

### Update all the packages in default project

Update-Package

### Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

### Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

#### For example:

```
'cmd
```

### Update specified Bold Reporting NuGet package

Update-Package BoldReports.Web

### Update specified package in default project with specified Package Source

Update-Package BoldReports.Web –Source “<https://api.nuget.org/v3/index.json>”

### Update specified package in specified project

Update-Package BoldReports.Web -ProjectName BoldReportingApplication

#### Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing nuget.exe to latest version use the following command:

update all NuGet packages from config file

Normal layout

```
`cmd
nuget update -self

```

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting tools NuGet packages.

```
`cmd
update all NuGet packages from config file
nuget update <configPath> [options]
```

```
update all NuGet packages from specified Packages Source
nuget update -Source <Source Location> [optional]

```

`configPath` is optional. It identifies the `package.config` or solutions file lists the packages utilized in the project.

**For example:**

```
`cmd
Update all NuGet packages from config file
nuget update "C:\Users\SyncfusionApplication\package.config"
```

```
Update all NuGet packages from specified Packages Source
nuget update -Source "https://api.nuget.org/v3/index.json"

```

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

## [Responsive layout rendering of ASP.NET Report Designer](#)

Report Designer will adaptively render itself with optimal user interfaces for tablet or desktop form factors. It has built-in responsive support, so that it will adjust automatically based on the browser viewport. Setting width to 100% is simply enough to make it responsive. This helps your application to scale elegantly on all form factors with ease.

### [Normal layout](#)

The following output shows the normal layout rendering of Report Designer tool bar items.

![Rendering of Report Designer tool bar items in normal layout](/static/assets/aspnet-web-forms/report-designer/images/normal-layout.png)

### [Responsive layout](#)

The following output shows the responsive layout rendering of Report Designer tool bar items.

![Rendering of Report Viewer tool bar items in responsive layout](/static/assets/aspnet-web-forms/report-designer/images/responsive-layout.png)

## Samples and Demos

Browse and explore our ready-to-use the designer samples, online and offline demos.

### Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

### Online demos

You can view the ASP.NET web forms Report Designer online demo samples from [here](#).

### GitHub demo samples

Click [here](#) to view the GitHub Report Designer demo samples.

## Supported Browsers

This document provides the list of web browsers supported by the Web Report Designer

| Desktop Browsers  | Version |
|-------------------|---------|
| Internet Explorer | 11.0+   |
| Microsoft Edge    | Current |
| Firefox           | 22+     |
| Google Chrome     | 17+     |
| Opera             | 12+     |
| Safari            | 5+      |

## How to queries for Bold Reports Report Designer

This section helps to get the answer for the frequently asked how to queries in Bold Reports Report Designer.

[Open server report using report path on opening Report Designer](#)

[How to add datasource and dataset for Report Designer from application?](#)

## How to open server report using report path at the time of opening the Report Designer

Find the following steps to open server report using report path on opening Report Designer.

Create a function and bind it with the `create` API in `Index.cshtml` file as like in below code snippet.

```
'html
<div style="height:500px">
```

```
@(Html.Bold().ReportDesigner("designer"))
.Create("controlInitialized")
)
</div>
<script type="text/javascript">
function controlInitialized(args) {
...
}
</script>
`
```

Use the `openReport` method in the function along with report path that was previously created, as like in below code snippet.

```
`javascript
function controlInitialized(args) {
var designer = $('#designer').data('boldReportDesigner');
designer.openReport("/Sample Reports/Sales Report");
}
`
```

## How to add the data source and dataset for Report Designer from application

You can add the data for reports in the Report Designer from application level on initializing the Report Designer. This can be achieved using the `addDataSource` and `addDataSet` functions, by which you can add the data source and dataset for the reports at the time of initialization of Report Designer.

Find the following steps to add the data source and dataset for Report Designer from application.

Create a function and bind it with the `create` API in `Index.cshtml` file as shown in the following code snippet.

```
`html
<div style="height:500px">
@(Html.Bold().ReportDesigner("designer"))
.Create("controlInitialized")
)
</div>
```

```
<script type="text/javascript">
function controlInitialized(args) {
...
}
</script>
`
```

Use the `addDatasource` method to add the data source in Report Designer as shown in the following code snippet,

```
'javascript
var datasource =
{
 type:'BoldReports.RDL.DOM.DataSource',
 Name:'DataSource1',
 Transaction:false,
 DataSourceReference:null,
 SecurityType:'DataBase',
 ConnectionProperties:{
 type:'BoldReports.RDL.DOM.ConnectionProperties',
 ConnectString:'Data Source=mvc.syncfusion.com;Initial Catalog=AdventureWorks;',
 EmbedCredentials:false,
 DataProvider:'SQL',
 IsDesignState:false,
 IntegratedSecurity:false,
 UserName:'',
 PassWord:'',
 Prompt:'Specify the Username and password for DataSource DataSource1',
 CustomProperties: []
 }
};
function controlInitialized(){
 var designerObj = $("#designer").data("boldReportDesigner");
 designerObj.addDataSource(datasource);
}
```

Use the `addDataSet` method to add dataset in Report Designer as shown in the following code snippet,

```
'javascript
var dataset =
{
 type:'BoldReports.RDL.DOM.DataSet',
 Name:'DataSet1',
 Fields:[
 { type: "BoldReports.RDL.DOM.Field", DataField: "DepartmentID", Name: "DepartmentID", TypeName: "System.Int16", Value: null},
 { type: "BoldReports.RDL.DOM.Field", DataField: "Name", Name: "Name", TypeName: "System.String", Value: null},
 { type: "BoldReports.RDL.DOM.Field", DataField: "GroupName", Name: "GroupName", TypeName: "System.String", Value: null },
 { type: "BoldReports.RDL.DOM.Field", DataField: "ModifiedDate", Name: "ModifiedDate", TypeName: "System.DateTime", Value: null}
],
 Query: {
 type: "BoldReports.RDL.DOM.Query",
 CommandText: "SELECT
[HumanResources].[Department].[DepartmentID],\n[HumanResources].[Department].[Name],\n[HumanResources].[Department].[GroupName],\n[HumanResources].[Department].[ModifiedDate] FROM
[HumanResources].[Department]",
 CommandType: 0,
 DataSourceName: "DataSource1",
 QueryDesignerState: {
 type: "BoldReports.RDL.DOM.QueryDesignerState",
 Expressions: null,
 Filters: null,
 Joins: null,
 StoredProcedure: null,
 Tables: [
 {
 type: "BoldReports.RDL.DOM.Table",

```

```
Columns: [
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
 IsSelected: true, Name: "DepartmentID"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
 IsSelected: true, Name: "Name"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
 IsSelected: true, Name: "GroupName"
},
{ type: "BoldReports.RDL.DOM.Column", AggregateTye: undefined, AliasName: null, IsDuplicate: false,
 IsSelected: true, Name: "ModifiedDate"
}
],
Name: "Department",
Schema: "HumanResources",
SchemaLevels: [
{ Name: "HumanResources", SchemaType: "Schema"},

{ Name: "Tables", SchemaType: "Category"},

{ Name: "Department", SchemaType: "Table"}
]
}
]
},
QueryParameters: [],
Timeout: 0
},
CaseSensitivity:0,
Collation:null,
AccentSensitivity:0,
KanatypeSensitivity:0,
WidthSensitivity:0,
Filters:[]
```

```

SharedDataSet:null,
InterpretSubtotalsAsDetails:0,
DataSetInfo:null,
DataSetObject:null
};

function controlInitialized(){
var designerObj = $("#designer").data("boldReportDesigner");
designerObj.addDataSet(dataset);
}
`
```

## Breaking Changes

This section provides the detailed information on API and behaviour changes that break existing applications when upgrading them to latest version of Bold Reports.

## Breaking Changes

When you upgrade from previous versions of Bold Reports to 2.2.23, there are few breaking changes. The following section explains the breaking changes for Bold Reports version 2.2.23.

### Designer resource read and write API

The resource write and read API's in `IReportDesignerController` are modified to simplify the resource write and read actions with Bold Report Designer.

Below are the new API details tabulated against old API's.

| Old API     | New API | Description                                         |
|-------------|---------|-----------------------------------------------------|
| UploadFile  | SetData | Writes the designer resource into storage location. |
| GetFilePath | GetData | Reads the designer resource from storage location.  |
| GetFiles    |         |                                                     |
| GetFile     |         |                                                     |

### Parameters

<span style="font-weight:bold">SetData</span>

| Parameter Name | Type   | Description                        |
|----------------|--------|------------------------------------|
| key            | string | Bold report designer unique token. |

|              |               |                |                                                                                                                                         |
|--------------|---------------|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| itemId       | string        |                | Unique id of designer resource.                                                                                                         |
| itemData     | object        |                | Gets the resource data.                                                                                                                 |
|              | Property Name | Type           |                                                                                                                                         |
|              | Data          | byte array     | Gets the resource data as byte array. To write an external resource into storage location, pass the resource data in this property.     |
|              | PostedFile    | HttpPostedFile | Gets the uploaded resource data as HttpPostedFile. The resource data uploaded within report designer will be received in this property. |
| errorMessage | string        |                | Returns the error message, if the write action is failed.                                                                               |

<span style="font-weight:bold">GetData</span>

| Parameter Name | Types  | Description                        |
|----------------|--------|------------------------------------|
| key            | string | Bold report designer unique token. |
| itemId         | string | Unique id of designer resource.    |

#### Return Type

| API Name | Return Type   |            |                                                          |
|----------|---------------|------------|----------------------------------------------------------|
| SetData  | boolean       |            |                                                          |
| GetData  | object        |            |                                                          |
|          | Property Name | Type       | Description                                              |
|          | Data          | byte array | Contains the requested resource data as byte array.      |
|          | errorMessage  | string     | Returns the error message, if the read action is failed. |

### Code snippet

| Old snippet                                                                                                                      | New snippet                                                                                                                                             |
|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>UploadFilecsharppublic bool UploadFile(System.Web.HttpPostedFile httpPostedFile){ //Implement resource write actions}</pre> | <pre>SetDatacsharppublic bool SetData(string key, string itemId, ItemInfo itemData, out string errorMessage){ //Implement resource write actions}</pre> |
| <pre>GetFilescsharppublic List GetFiles(FileType fileType){ //Implement resource read actions}</pre>                             |                                                                                                                                                         |
| <pre>GetFilePathcsharppublic string GetFilePath(string fileName){ //Implement actions to get file path}</pre>                    | <pre>GetDatacsharppublic ResourceInfo GetData(string key, string itemId){ //Implement resource read actions}</pre>                                      |
| <pre>GetFilecsharppublic FileModel GetFile(string filename, bool isOverride){ //Implement resource read actions}</pre>           |                                                                                                                                                         |

## Reporting tools for WPF

Enterprise-class reporting tools for WPF development to embed reporting functionalities such as viewing, exporting, and printing reports in your WPF applications.

### How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application.

### Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

## Reporting tools for WPF

Enterprise-class reporting tools for WPF development to embed reporting functionalities such as viewing, exporting, and printing reports in your WPF applications.

### How to best read this user guide

The best way to get started would be to read the [Getting Started](#) section of the documentation for the control that you would like to start using first. The [Getting Started](#) guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application.

### Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

## System Requirements

This topic describes the software and hardware requirements for setting up the development environment of Bold Reports WPF.

### Supported Operating Systems

Windows 7+, 8+

Windows Server 2008 R2+

### Hardware Requirements

The following hardware requirements are necessary for setting up the development environment of Bold Reports WPF:

1 GHZ or faster, 32 bit or 64 bit processor.

1 GB RAM for 32 bit or 2 GB RAM for 64 bit.

### Software Requirements

Microsoft Visual Studio 2010 or later

### Framework

The below tool is required for Bold Reports WPF.

.NET Framework 4.0 or higher

### See Also

[Licensing procedure for deployment](#)

## Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Microsoft Excel, CSV, Microsoft Word, and HTML formats. Some of the key features are,

Renders interactive reports with drill down, drill through, hyperlinks, and interactive sorting.

Easily customize each element of Report Viewer and provide events for report processing customization.

## Display ssrs rdl report in Bold Reports WPF Report Viewer

This section explains you the steps required to create your first WPF reporting application to display already created SSRS RDL report in Bold Reports WPF Report Viewer without using a Report Server.

### Create the application project

Open Visual Studio 2017 and select **File > New > Project**.

Go to **Installed > Visual C# > Windows Desktop**, and then select the required **.NET Framework** in the drop-down.

Select **WPF App (.NET Framework)**, change the application name, and then click **OK**.

![WPF application project template](/static/assets/wpf/report-viewer/images/getting-started/new-wpf-application.png)

Visual Studio creates the project and opens the designer for the default application window named **MainWindow.xaml**.

### Configure Report Viewer in an application

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**. Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution**.

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Wpf** NuGet package, and install them in your WPF application.

#### Package | Purpose

**BoldReports.Wpf** | Contains WPF Reporting controls (Report Viewer and Report Writer) to preview and export the reports.

### Initialize Report Viewer

Import the Report Viewer namespace as shown below in **MainWindow.xaml** file,

```
'csharp
xmlns:syncfusion="clr-namespace:BoldReports.UI.Xaml;assembly=BoldReports.Wpf"
'
```

Initialize Report Viewer component inside the **tag** as shown below in **MainWindow.xaml** file,

```
'csharp
<Window
.....
.....
.....
.....
xmlns:syncfusion="clr-namespace:BoldReports.UI.Xaml;assembly=BoldReports.Wpf"
.....
....>
<Grid>
<syncfusion:ReportViewer Name="reportViewer" />
</Grid>
</Window>
'
```

### Add already created reports

The Report Viewer is only for rendering the reports. You must use a report generation tool to create a report. To learn more about this, refer to the [create RDL report](#) section.

Create a folder **Resources** in your application to store the RDL reports.

Add already created reports to the newly created folder.

In this tutorial, the **sales-order-detail.rdl** report is used, and it can be downloaded in this [link](#). You can add the reports from Syncfusion installation location. For more information, refer to the [samples and demos](#) section.

Set the **Build Action** to **content** and **Copy to Output Directory** to either **Copy always** or **Copy if newer**.

![Build action template](/static/assets/wpf/report-viewer/images/getting-started/build-action.png)

### Set report path

Open **MainWindow.xaml.cs** file.

Initialize window loaded event inside the **MainWindow()** constructor.

```
'csharp
public MainWindow()
{

```

```
InitializeComponent();
this.Loaded += new RoutedEventHandler(MainWindow_Loaded);
}
private void MainWindow_Loaded(object sender, RoutedEventArgs e)
{
}
`
```

Set the `ReportPath` property in the `MainWindow_Loaded` event and invoke `RefreshReport()` method to render the report.

You can replace the following code in your `MainWindow_Loaded` event method.

```
'csharp
private void MainWindow_Loaded(object sender, RoutedEventArgs e)
{
 this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
 @"Resources\sales-order-detail.rdl");
 this.reportViewer.RefreshReport();
}
`
```

## Preview the report

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Preview of sales order detail report](/static/assets/wpf/report-viewer/images/getting-started/sales-order-detail.png)

## See Also

[Create RDLC report](#)

[Migrate Report Viewer](#)

[List of SSRS server versions are supported in Bold Reports](#)

## Load SSRS rdl reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports set the `ReportServerUrl` and `ReportServerCredential` properties as in the following code snippet.

```
'csharp
this.reportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.reportViewer.ReportServerCredential = new System.Net.NetworkCredential("ssrs", "RDLReport1");
```

```
this.reportViewer.ReportPath = "/SSRSSamples/Territory Sales"; //The report path should be in the
format of "/folder name/report name"
```

```
this.reportViewer.RefreshReport();
```

```
'
```

### Set data source credential for shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS server. If the report has any data source that uses credentials to connect with the database, then you must specify the **DataSourceCredentials** for each report data source to establish database connection.

```
'csharp
```

```
this.reportViewer.ReportLoaded += (sen, arg) =>
{
 List<BoldReports.Windows.DataSourceCredentials> dataSourceCrdentials = new
 List<BoldReports.Windows.DataSourceCredentials>();
 foreach (var dataSource in this.reportViewer.GetDataSources())
 {
 BoldReports.Windows.DataSourceCredentials crdentials = new
 BoldReports.Windows.DataSourceCredentials();
 crdentials.Name = dataSource.Name;
 crdentials.UserId = "ssrs1";
 crdentials.Password = "RDLReport1";
 dataSourceCrdentials.Add(crdentials);
 }
 this.reportViewer.SetDataSourceCredentials(dataSourceCrdentials);
};
this.reportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.reportViewer.ReportServerCredential = new System.Net.NetworkCredential("ssrs", "RDLReport1");
this.reportViewer.ReportPath = "/SSRSSamples/Territory Sales"; //The report path should be in the
format of "/folder name/report name"
this.reportViewer.RefreshReport();
'
```

Data source credentials must be added for shared data sources that do not have credentials in the connection strings.

### Render linked reports

You can render a linked report that point to an existing report, which is published in the SSRS Report Server. Also, it is possible to set the parameter, data source, credential, and other properties as like normal SSRS reports using the Report Viewer.

Load SharePoint integrated reports

Set data source credential for shared data sources

```
'csharp
this.reportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.reportViewer.ReportServerCredential = new System.Net.NetworkCredential("ssrs", "RDLReport1");
this.reportViewer.ReportPath = "/SSRSSamples/Territory Sales_Link"; //The report path should be in the
format of "/folder name/report name"
this.reportViewer.RefreshReport();
'
```

The **Territory Sales\_Link** is a linked report created for **Territory Sales** report that is already available on the SSRS Report Server.

## Load SharePoint integrated reports

To render SharePoint integrated SSRS reports set the **ReportPath**, **ReportServerUrl** and **ReportServerFormsCredential** properties as in the following code snippet.

```
'csharp
this.reportViewer.ReportPath =
"http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl";
this.reportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.reportViewer.ReportServerFormsCredential = new
BoldReports.Windows.ReportServerFormsCredential("ssrs", "RDLReport1");
this.reportViewer.RefreshReport();
'
```

## Set data source credential for shared data sources

The shared data source credentials can be added to the **DataSourceCredentials** property to connect with the database.

```
'csharp
this.reportViewer.ReportLoaded += (sen, arg) =>
{
 List<BoldReports.Windows.DataSourceCredentials> dataSourceCrdentials = new
 List<BoldReports.Windows.DataSourceCredentials>();
 foreach (var dataSource in this.reportViewer.GetDataSources())
 {
 BoldReports.Windows.DataSourceCredentials crdentials = new
 BoldReports.Windows.DataSourceCredentials();
 crdentials.Name = dataSource.Name;
 crdentials.UserId = "ssrs1";
 crdentials.Password = "RDLReport1";
 }
}
```

Load Bold report server reports

Set data source credential for shared data sources

```
dataSourceCrdentials.Add(credentials);
}

this.reportViewer.SetDataSourceCredentials(dataSourceCrdentials);
};

this.reportViewer.ReportPath =
"http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl";
this.reportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.reportViewer.ReportServerFormsCredential = new
BoldReports.Windows.ReportServerFormsCredential("ssrs", "RDLReport1");
this.reportViewer.RefreshReport();
`
```

Data source credentials must be added for shared data sources that do not have credentials in the connection strings.

## Load Bold report server reports

You can render the Bold Report Server hosted reports easily in your WPF report viewer application.

Add the `ReportServerExt.cs` and `DataClasses.cs` report server extension files in your WPF report viewer application to render the Bold Report server reports.

Download the `ReportServerExt.cs`, `DataClasses.cs` report server extension files from [here](#).

Set the Bold Report Server report properties as in following code snippet.

```
`csharp
if (System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())
{
 this.reportViewer.ReportServerUrl = @"https://on-premise-
demo.boldreports.com/reporting/api/site/site1/";
 this.reportViewer.ReportServer = new ReportingServerExt();
 this.reportViewer.ReportServerCredential = new
 System.Net.NetworkCredential("guest@boldreports.com", "demo");
 this.reportViewer.ReportPath = @"/Sample Reports/Company Sales";
 this.reportViewer.RefreshReport();
}
else
{`
```

```
MessageBox.Show("Internet connection is required to run this sample", "Report Server",
MessageBoxButton.OK);
}
```

## Render RDLC report

The data binding support, allows you to view RDLC reports that exist on the custom business object data collection. The following steps demonstrates how to render a RDLC report with custom business object data collection.

Add the RDLC report `product-list.rdlc` from Bold Reports installation location to your application `Resources` folder. For more information, see [Samples and demos](#).

Set the `ReportPath` and `ProcessingMode` to `ProcessingMode.Local` to the RDLC report location.

Bind the business object data values collection by adding new item to the `DataSources` as in the following code snippet.

```
'csharp
this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\product-list.rdlc");

this.reportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
this.reportViewer.DataSources.Clear();

this.reportViewer.DataSources.Add(new BoldReports.Windows.ReportDataSource { Name = "list", Value
= ProductList.GetData() });

this.reportViewer.RefreshReport();

.....
public class ProductList
{
 public string ProductName { get; set; }
 public string OrderId { get; set; }
 public double Price { get; set; }
 public string Category { get; set; }
 public string Ingredients { get; set; }
 public string ProductImage { get; set; }

 public static IList GetData()
 {
 List<ProductList> datas = new List<ProductList>();
```

```
ProductList data = null;
data = new ProductList()
{
 ProductName = "Baked Chicken and Cheese",
 OrderId = "323B60",
 Price = 55,
 Category = "Non-Veg",
 Ingredients = "grilled chicken, corn and olives.",
 ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
 ProductName = "Chicken Delite",
 OrderId = "323B61",
 Price = 100,
 Category = "Non-Veg",
 Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
 ProductImage = ""
};

datas.Add(data);
data = new ProductList()
{
 ProductName = "Chicken Tikka",
 OrderId = "323B62",
 Price = 64,
 Category = "Non-Veg",
 Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
 ProductImage = ""
};

datas.Add(data);
return datas;
}
```

```
}
```

### Load report as stream

To load report as a stream, create a report stream using the `FileStream` class and assign the report stream to the `Stream` property.

```
`csharp
```

```
FileStream reportStream = new FileStream(System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\product-list.rdlc"), FileMode.Open, FileAccess.Read);

this.reportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
this.reportViewer.LoadReport(reportStream);
this.reportViewer.DataSources.Clear();

this.reportViewer.DataSources.Add(new BoldReports.Windows.ReportDataSource { Name = "list", Value
= ProductList.GetData() });

this.reportViewer.RefreshReport();

....
```

```
public class ProductList
{
 public string ProductName { get; set; }
 public string OrderId { get; set; }
 public double Price { get; set; }
 public string Category { get; set; }
 public string Ingredients { get; set; }
 public string ProductImage { get; set; }

 public static IList GetData()
 {
 List<ProductList> datas = new List<ProductList>();
 ProductList data = null;
 data = new ProductList()
 {
 ProductName = "Baked Chicken and Cheese",
 OrderId = "323B60",
 Price = 55,
 Category = "Non-Veg",
 Ingredients = "grilled chicken, corn and olives."
 };
 datas.Add(data);
 return datas;
 }
}
```

```
ProductImage = "";
};
datas.Add(data);
data = new ProductList()
{
 ProductName = "Chicken Delite",
 OrderId = "323B61",
 Price = 100,
 Category = "Non-Veg",
 Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
 ProductImage = "";
};
datas.Add(data);
data = new ProductList()
{
 ProductName = "Chicken Tikka",
 OrderId = "323B62",
 Price = 64,
 Category = "Non-Veg",
 Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
 ProductImage = "";
};
datas.Add(data);
return datas;
}
}
`
```

## Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

Add the sub report and main reports to your application Resources folder. In this tutorial, using the already created reports. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section for creating new reports.

Render subreport

Load subreport stream

Download the `side-by-side-main-report.rdl`, `side-by-side-sub-report.rdl` reports from [here](#). Also, you can add the report from Syncfusion installation location. For more information, see [Samples and demos](#). The reports used from installed location, requires `NorthwindIO_Reports.sdf` database to run, so add it to your application.

Set the main report path `ReportPath` properties of the Report Viewer as in following code snippet.

```
'csharp
```

```
this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\side-by-side-main-report.rdl");

this.reportViewer.RefreshReport();

'
```

### Load subreport stream

To load subreport as stream, set the sub report stream in `LoadSubreport` method of the Report Viewer as in following code snippet.

```
'csharp
```

```
FileStream subReportStream = new FileStream(System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\product-list.rdlc"), FileMode.Open, FileAccess.Read);

FileStream mainReportStream = new
FileStream(System.IO.Path.Combine(Environment.CurrentDirectory, @"Resources\product-list-
main.rdlc"), FileMode.Open, FileAccess.Read);

this.reportViewer.SubreportProcessing += (sen, arg) =>
{
 this.reportViewer.DataSources.Clear();

 this.reportViewer.DataSources.Add(new BoldReports.Windows.ReportDataSource { Name = "list", Value
= ProductList.GetData() });

};

this.reportViewer.LoadSubreport("product-list", subReportStream);

this.reportViewer.LoadReport(mainReportStream);

this.reportViewer.RefreshReport();

.....

public class ProductList
{
 public string ProductName { get; set; }
 public string OrderId { get; set; }
 public double Price { get; set; }
 public string Category { get; set; }
}
```

Render subreport

Load subreport stream

```
public string Ingredients { get; set; }

public string ProductImage { get; set; }

public static IList GetData()

{

List<ProductList> datas = new List<ProductList>();

ProductList data = null;

data = new ProductList()

{

ProductName = "Baked Chicken and Cheese",

OrderId = "323B60",

Price = 55,

Category = "Non-Veg",

Ingredients = "grilled chicken, corn and olives.",

ProductImage = ""

};

datas.Add(data);

data = new ProductList()

{

ProductName = "Chicken Delite",

OrderId = "323B61",

Price = 100,

Category = "Non-Veg",

Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",

ProductImage = ""

};

datas.Add(data);

data = new ProductList()

{

ProductName = "Chicken Tikka",

OrderId = "323B62",

Price = 64,

Category = "Non-Veg",

Ingredients = "onions, grilled chicken, chicken salami & tomatoes."}
```

## Report parameters

## Set report parameter

```
ProductImage = ""
};
datas.Add(data);
return datas;
}
}
,
```

## Report parameters

Provides property options to pass or set report parameters default values at run-time using the `parameters` property. You can set the report parameters while creating the Report Viewer control in a application loaded.

In this tutorial, the `sales-order-dtail.rdl` report is used, and it can be downloaded from [here](#).

### Set report parameter

Set the default value data to the `Values`property and name of the report parameter to the `Name` property.

The parameter name is case sensitive, it should be same as available in the report definition.

The following code example illustrates how to set report parameter in the script.

```
'csharp
this.reportViewer.ReportLoaded += (sen, arg) =>
{
 List<BoldReports.Windows.ReportParameter> parameters = new
 List<BoldReports.Windows.ReportParameter>();
 BoldReports.Windows.ReportParameter parameter = new BoldReports.Windows.ReportParameter();
 parameter.Name = "SalesOrderNumber";
 parameter.Values = new List<string>() { "SO50756" };
 parameters.Add(parameter);
 this.reportViewer.SetParameters(parameters);
};
this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\sales-order-detail.rdl");
this.reportViewer.RefreshReport();
,
```

[Get report parameter](#)[Methods](#) | [Description](#)

GetParameters | Returns the parameters that are used in the current report without the processed values.

You can use the following code sample to get parameter names and set parameter default values.

'csharp

```
this.reportViewer.ReportLoaded += (sen, arg) =>
{
 var reportParameters = this.reportViewer.GetParameters();
 List<BoldReports.Windows.ReportParameter> parameters = new
 List<BoldReports.Windows.ReportParameter>();
 if (reportParameters != null)
 {
 foreach (var parameter in reportParameters)
 {
 parameters.Add(new BoldReports.Windows.ReportParameter()
 {
 Name = parameter.Name,
 Values = new List<string>() { "SO50756" }
 });
 }
 }
 this.reportViewer.SetParameters(parameters);
};

this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\sales-order-detail.rdl");
this.reportViewer.RefreshReport();
`
```

## Report interaction events

You can handle the report processing actions and interact with reports using the following events.

**ReportLoaded**

**ReportError**

**Drill through**

### Hyperlink

#### Report loaded

The **ReportLoaded** event fires once the report loading is completed and ready to start the processing. You can handle the event and specify data source, parameters at client-side. The following sample code loads a report by assigning report data source input in the **ReportLoaded** event.

In this tutorial, **product-list.rdlc** report is used, you can add the report from Bold Reports installation location. For more information, see [Samples and demos](#).

```
'csharp
this.reportViewer.ReportLoaded += (sen, arg) =>
{
 this.reportViewer.DataSources.Clear();
 this.reportViewer.DataSources.Add(new BoldReports.Windows.ReportDataSource { Name = "list", Value = ProductList.GetData() });
};

this.reportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\product-list.rdlc");
this.reportViewer.RefreshReport();
'
```

#### Report error

When an error occurs in the report processing, it raises the **ReportError** event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

```
'csharp
this.reportViewer.ReportError += (sen, arg) =>
{
 MessageBox.Show(arg.Message, "ReportError", MessageBoxButton.OK);
};

this.reportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
//this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\product-list.rdlc");
this.reportViewer.RefreshReport();
'
```

#### Drill through

When a drill through item is selected in a report, it invokes the **DrillThroughReport** event. You can change the drill through arguments such as report parameter and data sources. The following sample

code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

```
'csharp
this.reportViewer.DrillThroughReport += (sen, arg) =>
{
 List<BoldReports.Windows.ReportParameter> parameters = new
 List<BoldReports.Windows.ReportParameter>();
 BoldReports.Windows.ReportParameter parameter = new BoldReports.Windows.ReportParameter();
 parameter.Name = "EmployeeID";
 parameter.Values = new List<string>() { "4" };
 parameters.Add(parameter);
 //Assign the drill through report path and parameters
 arg.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory, @"Resources\personal-
details.rdl");
 arg.Parameters = parameters;
};

this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\sales-person-details.rdl");
this.reportViewer.RefreshReport();
'
```

## Hyperlink

The **Hyperlink** event occurs when the user clicks a hyperlink in a report, before the hyperlink is followed. The following sample code redirects to a new custom URL and cancels the component default action.

```
'csharp
this.reportViewer.Hyperlink += (sen, arg) =>
{
 arg.Cancel = true;
 System.Diagnostics.Process.Start(arg.Hyperlink);
};

this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\customer-support-analysis.rdl");
this.reportViewer.RefreshReport();
'
```

## Error logging in WPF Report Viewer

If an error occurred in report processing, WPF Report Viewer displays short messages about the error in view. In report viewer `ReportError` event raised. You can register the event and get the report error details from the event arguments to save all logs error information into a physical file location.

This section explains how to log the detailed error information to your WPF application.

This section requires a WPF Report Viewer application, if you don't have then create using the [Getting-Started](#).

In Solution Explorer, Open report viewer initialization cs file.

Register the `ReportError` event.

```
'csharp
private void MainWindow_Loaded(object sender, RoutedEventArgs e)
{
 this.reportViewer.ReportError += ReportViewer_ReportError;
}

private void ReportViewer_ReportError(object sender, ReportErrorEventArgs e)
{
 // You can register the report errors using event arguments.
}
```

Create a method in `MainWindow.xaml.cs` to write the error text into application folder.

```
'csharp
private void WriteLogs(string errorMessage)
{
 string filePath = System.IO.Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location) +
 @"\Errordetails.txt";

 using (StreamWriter writer = new StreamWriter(filePath, true))
 {
 writer.AutoFlush = true;
 writer.WriteLine(errorMessage);
 }
}
```

Invoke the newly created function in `ReportViewer_ReportError` as follows.

```
'csharp
private void ReportViewer_ReportError(object sender, ReportErrorEventArgs e)
{
 string errorLog;
 if (e.Exception != null)
 {
 errorLog = string.Format("Error Message: {0} \n Stack Trace: {1}", e.Message, e.Exception.StackTrace);
 }
 else
 {
 errorLog = e.Message;
 }
 WriteLogs(errorLog);
}
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final `MainWindow.xaml.cs` file is given as follows, you can replace it in your application.

```
'csharp
public partial class MainWindow : Window
{
 public MainWindow()
 {
 InitializeComponent();
 this.Loaded += new RoutedEventHandler(MainWindow_Loaded);
 }

 private void MainWindow_Loaded(object sender, RoutedEventArgs e)
 {
 this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
 @"Resources\sales-order-detail.rdl");
 this.ReportViewer.ReportError += ReportViewer_ReportError;
 }
}
```

```
this.reportViewer.RefreshReport();
}

private void ReportViewer_ReportError(object sender, ReportErrorEventArgs e)
{
 string errorLog;
 if (e.Exception != null)
 {
 errorLog = string.Format("Error Message: {0} \n Stack Trace: {1}", e.Message, e.Exception.StackTrace);
 }
 else
 {
 errorLog = e.Message;
 }
 WriteLogs(errorLog);
}

private void WriteLogs(string errorMessage)
{
 string filePath = System.IO.Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location) +
 @"\Errordetails.txt";
 using (StreamWriter writer = new StreamWriter(filePath, true))
 {
 writer.AutoFlush = true;
 writer.WriteLine(errorMessage);
 }
}
`
```

## Print report

The Report Viewer provides print option in the toolbar to print a copy of the report. The page setup dialog allows you to set the paper size or other page setup properties. To see print margins, click Print Layout on the toolbar.

The values allow you to set in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values for the page setup dialog come from the report properties, which are set in the design view.

## View report in print mode

Print margins are displayed in the Print Layout only, to view report in print mode by default, set the `ViewModel` property to `ViewModel.Print`.

```
'csharp
```

```
this.reportViewer.ViewMode = BoldReports.UI.Xaml.ViewModel.Print;
'
```

By default, the Report Viewer renders report in normal layout in which the print margins are not displayed.

## Remove empty spaces in printing

The extra blank page is created when the Body of your report is too wide for your page. If you want the report to appear on a single page, all the content within the report body must fit on the physical page and the body width should be lesser or equal to the following formula:

Body Width <= Page Width - (Left Margin + Right Margin)

For more details on designing a report to remove the empty pages in the report, refer to the knowledge base article of [report page sizing](#).

## Export report

The Report Viewer provides events and properties to control and customize the exporting functionality.

### PDF export options

The `PDFOptions` provides properties to manage PDF export behaviors. You have to set the properties in the application window loaded method.

#### Export with complex scripts

To export reports with the complex scripts, set the `EnableComplexScript` property of `PDFOptions` instance to true.

```
'csharp
```

```
this.reportViewer.PDFOptions = new BoldReports.Writer.PDFOptions()
{
 EnableComplexScript = true
};
'
```

### PDF Conformance

You can export the report as `PDF/A-1b` document by specifying the conformance level `PdfConformanceLevel.Pdf_A1B` in the `PdfConformanceLevel` property.

```
'csharp
```

```
this.reportViewer.PDFOptions = new BoldReports.Writer.PDFOptions()
{
 PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B
```

```
};
```

## Word export options

The **WordOptions** provides properties to manage Word document export behaviors.

### Word document type

You can save the report to the required document version by setting the **FormatType** property.

```
'csharp
```

```
this.reportViewer.WordOptions = new BoldReports.Writer.WordOptions();
```

```
this.reportViewer.WordOptions.FormatType = BoldReports.Writer.WordFormatType.Docx;
```

### Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the **LayoutOption** as **TopLevel**. The **ParagraphSpacing** is the distance value added between two elements in the document.

```
'csharp
```

```
this.reportViewer.WordOptions = new BoldReports.Writer.WordOptions();
```

```
this.reportViewer.WordOptions.LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel;
```

```
this.reportViewer.WordOptions.ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()
```

```
{
```

```
Bottom = 0.5f,
```

```
Top = 0.5f
```

```
};
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in word document is not a stand-alone object, if you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, added an empty paragraph between two tables.

### Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without a password. The following are the types of protection.

**AllowOnlyComments:** You can add or modify only the comments in the Word document.

**AllowOnlyFormFields:** You can modify the form field values in the Word document.

**AllowOnlyRevisions:** You can accept or reject the revisions in the Word document.

**AllowOnlyReading:** You can only view the content in the Word document.

**NoProtection:** You can access or edit the Word document contents as normally.

```
'csharp
```

```
this.reportViewer.WordOptions = new BoldReports.Writer.WordOptions();
this.reportViewer.WordOptions.ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading;
'
```

## Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

### Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

```
'csharp
```

```
this.reportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions();
this.reportViewer.ExcelOptions.ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013;
'
```

### Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the **LayoutOption** as **IgnoreCellMerge**.

```
'csharp
```

```
this.reportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions();
this.reportViewer.ExcelOptions.LayoutOption =
BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge;
'
```

### Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the **ExcelSheetProtection** or enabling the **ReadOnlyRecommended** properties.

```
'csharp
```

```
this.reportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
 ReadOnlyRecommended = true,
 ExcelSheetProtection = Syncfusion.XlsIO.ExcelSheetProtection.DeletingColumns
};
'
```

## PowerPoint export options

You can save the report to the required PowerPoint version by setting the **FormatType** property.

```
'csharp
this.reportViewer.PPTOptions = new BoldReports.Writer.PPTOptions();
this.reportViewer.PPTOptions.FormatType = BoldReports.Writer.PPTSaveType.PowerPoint2013;
'
```

### CSV export options

The `CsvOptions` allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
'csharp
this.reportViewer.CsvOptions = new BoldReports.Writer.CsvOptions()
{
 Encoding = System.Text.Encoding.Default,
 FieldDelimiter = ",",
 UseFormattedValues = false,
 Qualifier = "#",
 RecordDelimiter = "@",
 SuppressLineBreaks = true,
 FileExtension = ".txt"
};
'
```

### HTML export options

You can hide the separator added at the end of each page by setting the `HidePageSeparator` property to true.

```
'csharp
this.reportViewer.HTMLOptions = new BoldReports.Writer.HTMLOptions();
this.reportViewer.HTMLOptions.HidePageSeparator = true;
'
```

### Password protect exported document

This allows you to protect the exported document such as PDF, Word, Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet illustrates how to encrypt the exported document with the user defined password.

```
'csharp
//PDF encryption
this.reportViewer.PDFOptions = new BoldReports.Writer.PDFOptions();
this.reportViewer.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
{
```

```
UserPassword = "Password"
};
//Word encryption
this.reportViewer.WordOptions = new BoldReports.Writer.WordOptions()
{
 EncryptionPassword = "password"
};
//Excel encryption
this.reportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions()
{
 PasswordToModify = "password",
 PasswordToOpen = "password"
};
//PPT encryption
this.reportViewer.PPTOptions = new BoldReports.Writer.PPTOptions()
{
 EncryptionPassword = "password"
};
`
```

Password protection is not supported for HTML export format.

## Localization of Bold Report Viewer

Localization of Bold Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. Refer the following steps to localize the Report Viewer based on the culture.

Add Resources file for the different cultures.

Assign the value to each culture using key.

Assign a Current UI Culture to the application.

Set Report Viewer properties.

### Add the Resource file for the different cultures

Right-click the project and add the **Resources** folder in your application.

Right-click on the **Resources** folder and press **Ctrl+Shift+A** keys or select **Add > New Item** from the context menu.

In the Add New Item dialog, select **Resources** file and name it as **BoldReports.Wpf.resx**.

![Adding a new resource file](/static/assets/wpf/report-viewer/images/localization/add-new-resource-file.png)

Click **Add**.

In case, the another culture is used in the application, then create another resource file in name **[AssemblyName].[CultureInfo Code].resx** and the naming convention needs to be followed mandatorily.

### Assign the values to each culture using key

To assign **Values** in Resource, the resource file needs to be updated according to the following steps.

Open the **BoldReports.Wpf.resx** file by double clicking it from Solutions Explorer.

Add the resource key name, and its corresponding localized value by editing its field. Modify the resource values based on the **fr-FR** culture as shown in the following image.

![Edit resource file](/static/assets/wpf/report-viewer/images/localization/edit-resource-file.png)

You can download the modified resource file from [here](#) and replace it in your application.

### Assign a Current UI Culture to the application

Mention the culture to be referred while initializing the application, so that application refer to the appropriate value provided in resource file.

Set the **CultureInformation** in the application before the **InitializeComponent()** method call. In this application, **MainWindow.xaml.cs** is the startup page and culture is assigned as like below.

```
'csharp
public MainWindow()
{
 System.Globalization.CultureInfo.CurrentUICulture = new System.Globalization.CultureInfo("fr-FR");
 InitializeComponent();
}
'
```

### Set Report Viewer properties

Add the Report Viewer initialization code.

```
'csharp
```

|             |                   |
|-------------|-------------------|
| Limitations | RDL specification |
|-------------|-------------------|

```
this.reportViewer.ReportPath = System.IO.Path.Combine(Environment.CurrentDirectory,
@"Resources\sales-order-detail.rdl");
```

```
this.reportViewer.RefreshReport();
```

In this tutorial, sales-order-detail report is used.

Now, run the application and the below output shows the toolbar items localized fr-FR culture.

![Localization output](/static/assets/wpf/report-viewer/images/localization/localization-output.png)

## Limitations

### RDL specification

The Report Viewer control does not support RDL specification for SQL Server 2000 and SQL Server 2005.

### Report layout

In the Tablix cell split layout process, the entire cell moves to the next page to display the complete cell items, when the table cell width value exceeds the page width.

### Expressions

The object function and VB function do not have complete support.

### SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you should specify the data source credentials for each report data source to establish database connection.

## Samples and Demos

Browse and explore the ready-to-use RDL, RDLC reports, samples, and offline demos.

### Locally installed reports

You can obtain the sample RDL and RDLC files from the Bold Reports installed location

%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.

### Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

## Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of WPF Report Viewer application:

## Client-side migration

In the Solution Explorer, right-click the **References** and remove the following assembly references:

Syncfusion.ReportControls.Wpf

Syncfusion.ReportViewer.Wpf

Syncfusion.ReportWriter.Base

Add the assembly from the Bold Reporting NuGet package **BoldReports.Wpf**. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for **BoldReports.Wpf** NuGet package, and install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

### Control initialization

Open the **MainWindow.xaml** file and remove the following namespace in your XAML page.

```
'csharp
xmlns:syncfusion="http://schemas.syncfusion.com/wpf"
'
```

Add the following namespace in your XAML page.

```
'csharp
xmlns:syncfusion="clr-namespace:BoldReports.UI.Xaml;assembly=BoldReports.Wpf"
'
```

## NuGet Packages for WPF

Refer to the following steps to configure Bold Reporting NuGet packages for WPF application.

### Configure NuGet feed URL

#### Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** NuGet.org

**Source:** <https://api.nuget.org/v3/index.json>

![Online NuGet Configure] (/static/assets/wpf/report-viewer/images/nuget-packages/reporting-online-nuget-packages-configuration.png)

### Installing NuGet packages

#### Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**.

Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the **BoldReports.WPF** package, and then click **Install** button.

#### Install using Package Manager Console

To install the Bold Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

#### install specified package in default project

```
Install-Package <Package Name>
```

#### install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

#### install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

```
'
```

#### For example:

```
'cmd
```

#### install specified package in default project

```
Install-Package BoldReports.Wpf
```

## install specified package in default project with specified Package Source

Install-Package BoldReports.Wpf –Source “<https://api.nuget.org/v3/index.json>”

## install specified package in specified project

Install-Package BoldReports.Wpf -ProjectName BoldReportsApplication

## Upgrading NuGet packages

### Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**

Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

### Upgrading using Package Manger Console

To update the installed Bold Reporting NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

`cmd

## Update specific NuGet package in default project

Update-Package <Package Name>

## Update all the packages in default project

Update-Package

## Update specified package in default project with specified package source

Update-Package <Package Name> -Source <Source Location>

## Update specified package in specified project

Update-Package <Package Name> - ProjectName <Project Name>

**For example:**

```
`cmd
```

## Update specified Bold Reporting NuGet package

```
Update-Package BoldReports.Wpf
```

## Update specified package in default project with specified Package Source

```
Update-Package BoldReports.Wpf -Source "https://api.nuget.org/v3/index.json"
```

## Update specified package in specified project

```
Update-Package BoldReports.Wpf -ProjectName BoldReportsApplication
```

```
`
```

## Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing nuget.exe to latest version use the following command:

```
`cmd
```

```
nuget update -self
```

```
`
```

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting NuGet packages.

```
`cmd
```

## update all NuGet packages from config file

```
nuget update <configPath> [options]
```

## update all NuGet packages from specified Packages Source

```
nuget update -Source <Source Location> [optional]
```

```
`
```

**configPath** is optional. It identifies the **package.config** or solutions file lists the packages utilized in the project.

**For example:**

```
`cmd
```

## Update all NuGet packages from config file

```
nuget update "C:\Users\BoldReportsApplication\package.config"
```

## Update all NuGet packages from specified Packages Source

nuget update -Source "https://api.nuget.org/v3/index.json"

\

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

## How to Create RDL/RDLC Report

The following sections explain about how to create a new RDL/RDLC report using Bold Report Designer, Microsoft Report Builder and Visual Studio Report Server project template.

[Create a RDL report](#)

[Create a RDLC report](#)

## Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

Bold Reports Report Designer.

Microsoft Report Builder.

Visual Studio Report Server project template.

## Bold Reports Report Designer

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

## Microsoft SQL Report Builder

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

## Visual Studio Report Server template

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

## Create a RDLC report using business object data source

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

## Prerequisites

Microsoft Visual Studio 2017 or higher

### [Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

## Create business object class

Open Visual Studio from the File menu and select **New Project**.

Create project with class library type from the project type list.

![Add a new class library project](/static/assets/wpf/report-viewer/images/how-to/create-report/class-library-project.png)

Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
 public string ProdCat { get; set; }
 public string SubCat { get; set; }
 public string OrderYear { get; set; }
 public string OrderQtr { get; set; }
 public double Sales { get; set; }
}
```

Clean and build the application.

## Add an RDLC report

Right-click the project and click **Add > New Item**.

Search Report with new item and select **Report Wizard** to start the report creation with dataset selection.

Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/wpf/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

## Data source and table configuration wizard

Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/wpf/report-viewer/images/how-to/create-report/choose-data-source-type.png)

Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/wpf/report-viewer/images/how-to/create-report/select-data-objects.png)

In the DataSet Properties wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/wpf/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/wpf/report-viewer/images/how-to/create-report/arrange-table-fields.png)

Choose the table layout and click **Next**.

Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/wpf/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/wpf/report-viewer/images/how-to/create-report/sales-report-design.png)

## Bold Report Writer

Report Writer is a class library that enables the user to render reports defined in Microsoft's RDL format (2008 or 2008 R2) as PDF, Word, Excel or CSV documents.

The important features of WPF Report Writer are listed as follows:

RDL Specification - Supports RDL specification for the SQL Server 2008 and RDL specification for the SQL Server 2008 R2 only. List of available report definition formats:  
[msdn.microsoft.com/library/dd297486\(SQL.100\).](https://msdn.microsoft.com/library/dd297486(SQL.100).)

Data sources - You can use advanced database servers Data Sources in Report Writer (SQL and Oracle).

Charts - Show all basic types of charts that are available in Microsoft RDL reports.

Tablix - Shows the summaries and simple tables.

Gauge - Shows measurement values by using expression values.

Textbox - Shows textbox data with expression support.

Export - Export report as PDF, Word, Excel and CSV.

Report Parameter - Views the report based on the report parameter value.

## Export SSRS RDL Report in Bold Reports WPF Report Writer

The Report Writer is a class library that is used to export the RDL report with popular file formats like PDF, Microsoft Word, Microsoft CSV, and Microsoft Excel without previewing the report in webpage. This section describes how to export the RDL report in WPF application using the **Report Writer**.

### Create WPF application

Start Visual Studio 2019 and click **Create new project**.

Choose **WPF App (.NET Framework)**, and then click **Next**.

![WPF application project template](/static/assets/wpf/report-writer/images/getting-started/new-wpf-application.png)

Change the project name, and then click **Create**.

### List of dependency libraries

In the Solution Explorer tab right-click the project or solution , and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **BoldReports.Wpf** package, and install this in your WPF application. The following table provides details about the packages and their usage.

### Package | Purpose

**BoldReports.Wpf** | Contains WPF Reporting controls (Report Viewer and Report Writer) to preview and export the reports.

### Add already created reports

In this tutorial, the **sales-order-detail.rdl** report is used, and it can be downloaded in this [link](#). You can add the reports from Syncfusion installation location. For more information, refer to the [samples and demos](#) section.

Create a folder **Resources** in your application to store the RDL reports.

Add already created reports to the newly created folder.

## Initialize Report Writer

Open the **MainWindow.xaml** file in your application. Add the following code example in the **tag**.

```
'csharp
<Window
.....
.....
.....
Title="WPF Writer" WindowStyle="SingleBorderWindow" ResizeMode="NoResize"
WindowStartupLocation="CenterScreen" Width="365" Height="235">
<Grid>
</Grid>
</Window>
'
```

Initialize Report writer export type inside the **tag** as shown below in **MainWindow.xaml** file,

```
'csharp
<Grid>
<Grid.RowDefinitions>
<RowDefinition Height="*"/>
<RowDefinition Height="*"/>
</Grid.RowDefinitions>
<TextBlock Grid.Row="0" TextAlignment="Justify" FontFamily="Verdana" FontSize="11"
TextWrapping="Wrap" Padding="5" >
<TextBlock.Background>
<LinearGradientBrush EndPoint="0.5,-0.04" StartPoint="0.5,1.04">
<GradientStop Color="#FFD9E9F7" Offset="0"/>
<GradientStop Color="#FFEFFF" Offset="1"/>
</LinearGradientBrush>
</TextBlock.Background>
<TextBlock.Text>
```

Choose a file format to view the selected document generated from Report file by using Essential Report Writer.

```

</TextBlock.Text>
</TextBlock>
<StackPanel Grid.Row="1" Orientation="Horizontal">
<RadioButton Content="PDF" Height="16" HorizontalAlignment="Left" Name="pdf" IsChecked="True" Margin="5"/>
<RadioButton Content="Excel" Height="16" HorizontalAlignment="Left" Name="excel" Margin="5" />
<RadioButton Content="Word" Height="16" HorizontalAlignment="Left" Name="word" Margin="5"/>
<RadioButton Content="HTML" Height="16" HorizontalAlignment="Left" Name="html" Margin="5"/>
<Button Click="Button_Click" Margin="10,0" Width="100" Height="25" BorderBrush="LightBlue">
<Button.Background>
<LinearGradientBrush EndPoint="0.5,-0.04" StartPoint="0.5,1.04">
<GradientStop Color="#FFD9E9F7" Offset="0"/>
<GradientStop Color="#FFE9F7E9" Offset="1"/>
</LinearGradientBrush>
</Button.Background>
<StackPanel Orientation= "Horizontal">
<Image Name="image2" Margin="2" />
<TextBlock Text="Generate" Margin="3" HorizontalAlignment="Right" />
</StackPanel>
</Button>
</StackPanel>
</Grid>
`
```

### [Set report path](#)

Open `MainWindow.xaml.cs` file and add the following using statement.

```

`csharp
using BoldReports.Writer;
`
```

Initialize ReportWriter by using the following code example in the `MainWindow.xaml.cs` file export button click event.

```
`csharp
private void Button_Click(object sender, RoutedEventArgs e)
{
try
{
// Refer the RDL report file location
string reportPath = @"<root folder>\Resources\sales-order-detail.rdl";
string fileName = null;
WriterFormat format;
//Step 1 : Instantiate the report writer with the parameter "ReportPath".
ReportWriter reportWriter = new ReportWriter(reportPath);
//Step 2 : Save the report as Pdf or Word or Excel
if (pdf.IsChecked == true)
{
fileName = "sales-order-detail.pdf";
format = WriterFormat.PDF;
}
else if (word.IsChecked == true)
{
fileName = "sales-order-detail.doc";
format = WriterFormat.Word;
}
else if (excel.IsChecked == true)
{
fileName = "sales-order-detail.xls";
format = WriterFormat.Excel;
}
else
{
fileName = "sales-order-detail.html";
format = WriterFormat.HTML;
}
reportWriter.Save(fileName, format);
}
```

```
//Message box confirmation to view the created report document.
if (MessageBox.Show("Do you want to view the " + format + " file?", "" + format + " report Created",
MessageBoxButton.YesNo, MessageBoxIcon.Information) == MessageBoxResult.Yes)
{
 //Launching the PDF file using the default Application.[Acrobat Reader]
 System.Diagnostics.Process.Start(fileName);
}
}
}
catch (Exception Ex)
{
 MessageBox.Show(Ex.Message);
}
}
}
`
```

Now, Run and export the report with specified export format in your Report Writer application.

Congratulations! you have completed your first WPF Writer application!.Click [here](#) to download the already created WPF Report Writer application.

Exported files are saved in application bin location.

## Frequently asked questions

This section helps to get the answer for the frequently asked questions. Discussed the following topic in this section.

[How to print the report silently without preview the report in Report Viewer?](#)

## How to print the report silently without preview the report in Report Viewer

You have to use the **Report Writer** and **PdfDocumentView** to print the reports silently without viewing in Report Viewer. Find the following steps to print the report silently:

Initialize the [Report Writer](#) with reports and data sources, use ~Save~ API to get the printable PDF in stream.

```
'csharp
string reportPath = @"../../ReportTemplate/Product Details.rdlc";
ReportWriter reportWriter = new ReportWriter(reportPath);
reportWriter.ReportProcessingMode = ProcessingMode.Local;
```

```
reportWriter.DataSources.Clear();
reportWriter.DataSources.Add(new ReportDataSource { Name = "DataSet1", Value =
ProductCatalog.GetData() });

MemoryStream stream = new MemoryStream();
reportWriter.Save(stream, WriterFormat.PDF);
`
```

Load the exported stream into the PdfDocumentView using the **PdfDocumentView.Load** API and add the available printers to a list for printing the report using the Windows PrintDialog as shown in the following code example.

```
`csharp
PdfDocumentView pdfdoc = new PdfDocumentView();
pdfdoc.Load(stream);

var doc = pdfdoc.PrintDocument as IDocumentPaginatorSource;
PrintDialog printDialog = new PrintDialog();
List<string> printersList = new List<string>();
List<string> serversList = new List<string>();
var server = new PrintServer();
var queues = server.GetPrintQueues(new[] { EnumeratedPrintQueueTypes.Local });
foreach (var queue in queues)
{
if (!serversList.Contains(queue.HostingPrintServer.Name))
{
serversList.Add(queue.HostingPrintServer.Name);
}
printersList.Add(queue.FullName);
}
server = new PrintServer(serversList[0].ToString());
PrintQueue printer1 = server.GetPrintQueue(printersList[2].ToString());
printDialog.PrintQueue = printer1;
printDialog.PrintDocument(doc.DocumentPaginator, "PDF PRINTER");
`
```

## Reporting tools for UWP

Enterprise-class reporting tools for UWP development to embed reporting functionalities such as viewing, exporting, and printing reports in your UWP applications.

## How to best read this user guide

The best way to get started would be to read the Getting Started section of the documentation for the control that you would like to start using first. The Getting Started guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application.

### Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

## Reporting tools for UWP

Enterprise-class reporting tools for UWP development to embed reporting functionalities such as viewing, exporting, and printing reports in your UWP applications.

### How to best read this user guide

The best way to get started would be to read the Getting Started section of the documentation for the control that you would like to start using first. The Getting Started guide gives just enough information that you need to know before starting to write code. This is the only section that we recommend reading end-to-end before starting to write code, all other information can be referred as needed.

Now that you are familiar with the basics of using the control, the next step would be to start integrating the control into your application.

### Getting help

If you are still not able to find the information that you are looking for in the self-help resources mentioned above then please [contact us](#) by creating a support ticket.

## System Requirements

This topic describes the software and hardware requirements for setting up the development environment of Bold Reports UWP.

### Supported Operating Systems

Windows 7+, 8+

Windows Server 2008 R2+

### Hardware Requirements

The following hardware requirements are necessary for setting up the development environment of Bold Reports UWP:

1 GHZ or faster, 32 bit or 64 bit processor.

1 GB RAM for 32 bit or 2 GB RAM for 64 bit.

## Software Requirements

Microsoft Visual Studio 2010 or later

### Framework

The below tool is required for Bold Reports UWP.

.NET Framework 4.0 or higher

### See Also

[Licensing procedure for deployment](#)

## Overview

The Report Viewer is a visualization control used to display SSRS, RDL, RDLC, and Bold Report Server reports within web applications. It allows you to view RDL/RDLC reports with or without using SSRS or Bold Report Server. You can bind data sources, parameters, and render reports with all major capabilities of RDL reporting and export the report to PDF, Microsoft Excel, Microsoft Word, and HTML formats.

## Display ssrs rdl report in Bold Reports UWP Report Viewer

This section explains you the steps required to create your first UWP reporting application to display already created SSRS RDL report in Bold Reports UWP Report Viewer without using a Report Server.

### Create the application project

Open Visual Studio 2017 and select **File > New > Project**.

Go to **Installed > Visual C# > Windows Universal**.

Select **Blank App (Universal Windows)**, change the application name, and then click **OK**.

![UWP application project template](/static/assets/uwp/report-viewer/images/getting-started/new-uwp-application.png)

Select the target and minimum platform version **Windows 10, version 1809(10.0; Build 17763)** from the dropdown, and then click **OK**.

![New universal windows platform project](/static/assets/uwp/report-viewer/images/getting-started/new-universal-windows-platform-project.png)

Set same version for the target and minimum platform version.

### Configure Report Viewer in an application

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution**.

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.UWP** NuGet package, and install them in your UWP application.

#### Package | Purpose

**BoldReports.UWP** | Contains UWP Reporting controls (Report Viewer and Report Writer) to preview and export the reports.

#### [Initialize Report Viewer](#)

Open **MainWindow.xaml** file and import the Report Viewer namespace as shown below,

```
'csharp
xmlns:BoldReports="using:BoldReports.UI.Xaml"
'
```

Initialize Report Viewer component inside the **tag** as shown below in **MainWindow.xaml** file,

```
'csharp
<Page
.....
.....
.....
xmlns:BoldReports="using:BoldReports.UI.Xaml"
....>
<Grid>
<BoldReports:ReportViewer Name="ReportViewer"/>
</Grid>
</Page>
'
```

#### [Create Web API service](#)

The Report Viewer requires a Web API service to process the RDL report files. You can skip this step and use the online [Web API services](#) to preview the already available reports or you should create any one of the following Web API services:

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

### Adding already created report

If you have created a new service, you can add the reports from the Syncfusion installation location. For more information, refer to [samples and demos](#) section.

Create a folder **Resources** in your Web API application to store RDL reports and add the already created reports to it.

Add already created reports to the newly created folder.

In this tutorial, the **sales-order-detail.rdl** report is used, and it can be downloaded in this [link](#).

Refer to the [create RDL report](#) section for creating new reports.

### Set report path and service URL

Open **MainWindow.xaml.cs** file.

Initialize window loaded event inside the **MainWindow()** constructor.

```
'csharp
public MainPage()
{
 this.InitializeComponent();
 this.Loaded += MainPage_Loaded;
}

private void MainPage_Loaded(object sender, RoutedEventArgs e)
{
}
```

Set the **ReportPath** and **ReportServiceUrl** properties in the **MainWindow\_Loaded** event method and invoke **RefreshReport()** method to render the report.

You can replace the following code in your **MainWindow\_Loaded** event method.

```
'csharp
private void MainPage_Loaded(object sender, RoutedEventArgs e)
{
 this.ReportViewer.ReportPath = "~/Resources/docs/sales-order-dtail.rdl";
 this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
 this.ReportViewer.RefreshReport();
```

```
}
```

In the above code, the `sales-order-detail.rdl` report and `reportServiceUrl` used from online URL.

### Preview the report

Build and run the application to view the report output in the Report Viewer as displayed in the following screenshot.

![Preview of sales order detail report](/static/assets/uwp/report-viewer/images/getting-started/sales-order-detail.png)

### See Also

[Create RDLC report](#)

[Migrate Report Viewer](#)

[List of SSRS server versions are supported in Bold Reports](#)

## Load SSRS Report Server reports

Report Viewer has support to load RDL reports from SSRS Report Server. To render SSRS Reports set the `ReportServerUrl` and `ReportServerCredential` properties as in the following code snippet.

```
'csharp
```

```
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.ReportViewer.ReportServerCredential = new System.Net.NetworkCredential("ssrs", "RDLReport1");
this.ReportViewer.ReportPath = "/SSRSSamples/Territory Sales"; //The report path should be in the
format of "/folder name/report name"
this.ReportViewer.RefreshReport();
```

### Set data source credential for shared data sources

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the SSRS server. If the report has any data source that uses credentials to connect with the database, then you must specify the `DataSourceCredentials` for each report data source to establish database connection.

```
'csharp
```

```
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.ReportViewer.ReportPath = "/SSRSSamples/Territory Sales"; //The report path should be in the
format of "/folder name/report name"
this.ReportViewer.RefreshReport();
```

‘

To set shared datasource credentials in Web API Controller, use the following code in the **OnReportLoaded** method.

`csharp

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
 //Add SSRS Report Server and data source credentials
 reportOption.ReportModel.ReportServerCredential = new System.Net.NetworkCredential("ssrs",
 "RDLReport1");

 reportOption.ReportModel.DataSourceCredentials.Add(new
 BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));

}
```

‘

## Render linked reports

You can render a linked report that point to an existing report, which is published in the SSRS Report Server. Also, it is possible to set the parameter, data source, credential, and other properties as like normal SSRS reports using the Report Viewer.

`csharp

```
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.ReportViewer.ReportServerCredential = new System.Net.NetworkCredential("ssrs", "RDLReport1");
this.ReportViewer.ReportPath = "/SSRSSamples/Territory Sales_Link"; //The report path should be in
the format of "/folder name/report name"
this.ReportViewer.RefreshReport();
```

‘

The **Territory Sales\_Link** is a linked report created for **Territory Sales** report that is already available on the SSRS Report Server.

## Load SharePoint Server reports

To render SharePoint integrated SSRS reports set the **ReportPath**, **ReportServiceURL**, **ReportServerUrl** and **ReportServerFormsCredential** properties as in the following code snippet.

`csharp

```
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.ReportViewer.ReportServerFormsCredential = new
BoldReports.UI.Xaml.ReportServerFormsCredential("ssrs", "RDLReport1");
```

Load Bold Report Server reports

Set data source credential for shared data sources

```
this.ReportViewer.ReportPath =
"http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl";
this.ReportViewer.RefreshReport();
'
```

## Set data source credential for shared data sources

The shared data source credentials can be added to the **DataSourceCredentials** property to connect with the database.

```
'csharp
```

```
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportServerUrl = "http://<servername>/reportserver$instanceName";
this.ReportViewer.ReportPath =
"http://<servername>/reportserver$instanceName/SSRSSamples/Territory Sales.rdl";
this.ReportViewer.RefreshReport();
'
```

To set shared datasource credentials in Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
```

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
//Add ReportServerFormsCredential and data source credentials
reportOption.ReportModel.ReportServerFormsCredential = new
BoldReports.Web.ReportServerFormsCredential("ssrs", "RDLReport1");
reportOption.ReportModel.DataSourceCredentials.Add(new
BoldReports.Web.DataSourceCredentials("AdventureWorks", "ssrs1", "RDLReport1"));
}
```

Data source credentials must be added for shared data sources that do not have credentials in the connection strings.

## Load Bold Report Server reports

You can render the Bold Report Server hosted reports in the Report Viewer easily without creating a Web API service. Bold Report Server provides the built-in Web API service that helps you to display the server reports.

Set the Bold Report Server built-in service URL `ReportServiceURL` and `ReportServerUrl`, `ReportServerCredential`, `ReportPath` to a report deployed on the server. You can use the following complete code in your application to render the Bold Report Server reports.

```
'csharp

this.ReportViewer.ReportServiceURL = @"https://on-premise-
demo.boldreports.com/reporting/reportservice/api/Viewer/";

this.ReportViewer.ReportServerUrl = @"https://on-premise-
demo.boldreports.com/reporting/api/site/site1";

this.ReportViewer.ServiceAuthorizationToken = GenerateToken("guest@boldreports.com", "demo");

this.ReportViewer.ReportServiceRequestBegin += (sen, arg) =>

{

 arg.HttpClient.DefaultRequestHeaders.Add("serverurl", "https://on-premise-
 demo.boldreports.com/reporting/api/site/site1");

};

this.ReportViewer.ReportPath = @"/Sample Reports/Company Sales";
this.ReportViewer.RefreshReport();

public string GenerateToken(string userName, string password)
{
 using (var client = new HttpClient())
 {
 client.DefaultRequestHeaders.Accept.Clear();

 var content = new FormUrlEncodedContent(new[]
 {

 new KeyValuePair<string, string>("grant_type", "password"),
 new KeyValuePair<string, string>("username", userName),
 new KeyValuePair<string, string>("password", password)
 });

 var result = client.PostAsync("https://on-premise-
 demo.boldreports.com/reporting/api/site/site1/token", content).Result;

 string resultContent = result.Content.ReadAsStringAsync().Result;

 var token = JsonConvert.DeserializeObject<Token>(resultContent);

 return token.tokenType + " " + token.accessToken;
 }
}
```

```
public class Token
{
 public string access_token { get; set; }
 public string token_type { get; set; }
 public string expires_in { get; set; }
 public string userName { get; set; }
 public string serverUrl { get; set; }
 public string password { get; set; }
}
```

## Render RDLC report

The data binding support, allows you to view RDLC reports that exist on the custom business object data collection. The following steps demonstrates how to render a RDLC report with custom business object data collection.

Add the RDLC report `product-list.rdlc` from Bold Reports installation location to your application Resources folder. For more information, see [Samples and demos](#).

Assign the `processingMode` property to `ProcessingMode.Local`.

To load report as a stream, create a report stream using the `Stream` class and assign the report stream to the `LoadReport()` method.

Bind the business object data values collection by adding new item to the `DataSources` as in the following code snippet.

```
'csharp
Assembly assembly = typeof(MainPage).GetTypeInfo().Assembly;
Stream reportStream = assembly.GetManifestResourceStream("ReportViewer.Resources.product-
list.rdlc");
this.ReportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
this.ReportViewer.LoadReport(reportStream);
this.ReportViewer.DataSources.Clear();
this.ReportViewer.DataSources.Add(new BoldReports.UI.Xaml.ReportDataSource { Name = "list", Value
= ProductList.GetData() });
this.ReportViewer.RefreshReport();
.....
public class ProductList
```

```
{
 public string ProductName { get; set; }
 public string OrderId { get; set; }
 public double Price { get; set; }
 public string Category { get; set; }
 public string Ingredients { get; set; }
 public string ProductImage { get; set; }
 public static IList GetData()
 {
 List<ProductList> datas = new List<ProductList>();
 ProductList data = null;
 data = new ProductList()
 {
 ProductName = "Baked Chicken and Cheese",
 OrderId = "323B60",
 Price = 55,
 Category = "Non-Veg",
 Ingredients = "grilled chicken, corn and olives.",
 ProductImage = ""
 };
 datas.Add(data);
 data = new ProductList()
 {
 ProductName = "Chicken Delite",
 OrderId = "323B61",
 Price = 100,
 Category = "Non-Veg",
 Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
 ProductImage = ""
 };
 datas.Add(data);
 data = new ProductList()
 {
```

Render subreport

Load subreport stream

```
ProductName = "Chicken Tikka",
OrderId = "323B62",
Price = 64,
Category = "Non-Veg",
Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
ProductImage = ""
};

datas.Add(data);
return datas;
}

`
```

## Render subreport

You can display another report inside the body of a main report using the Report Viewer. The following steps helps you to customize the subreport properties such as data source, report path, and parameters.

Add the sub report and main reports to your application Resources folder. In this tutorial, using the already created reports. Refer to the [Create RDL Report section](#) or [Create RDLC Report section](#) section for creating new reports.

Download the `side-by-side-main-report.rdl`, `side-by-side-sub-report.rdl` reports from [here](#). Also, you can add the report from Syncfusion installation location. For more information, see [Samples and demos](#). The reports used from installed location, requires `NorthwindIO_Reports.sdf` database to run, so add it to your application.

Set the main report path `ReportPath` and `ReportServiceURL` properties of the Report Viewer as in following code snippet.

```
'csharp

this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportPath = "~/Resources/docs/side-by-side-main-report.rdl";
this.ReportViewer.RefreshReport();
`
```

## Load subreport stream

To load subreport as stream, set the sub report stream in `LoadSubreport()` method of the Report Viewer as in following code snippet.

```
'csharp
```

Render subreport

Load subreport stream

```
Assembly assembly = typeof(MainPage).GetTypeInfo().Assembly;
Stream mainReportStream =
assembly.GetManifestResourceStream("ReportViewer_UWP.Resources.product-list-main.rdlc");
Stream subReportStream =
assembly.GetManifestResourceStream("ReportViewer_UWP.Resources.product-list.rdlc");
this.ReportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
this.ReportViewer.SubreportProcessing += (sen, arg) =>
{
 this.ReportViewer.DataSources.Clear();
 this.ReportViewer.DataSources.Add(new BoldReports.UI.Xaml.ReportDataSource { Name = "list", Value = ProductList.GetData() });
};
this.ReportViewer.LoadSubreport("product-list", subReportStream);
this.ReportViewer.LoadReport(mainReportStream);
this.ReportViewer.RefreshReport();
.....
public class ProductList
{
 public string ProductName { get; set; }
 public string OrderId { get; set; }
 public double Price { get; set; }
 public string Category { get; set; }
 public string Ingredients { get; set; }
 public string ProductImage { get; set; }
 public static IList GetData()
 {
 List<ProductList> datas = new List<ProductList>();
 ProductList data = null;
 data = new ProductList()
 {
 ProductName = "Baked Chicken and Cheese",
 OrderId = "323B60",
 Price = 55,
 Category = "Non-Veg",
 };
 datas.Add(data);
 return datas;
 }
}
```

Report parameters

Load subreport stream

```
Ingredients = "grilled chicken, corn and olives.",
ProductImage = ""
};
datas.Add(data);
data = new ProductList()
{
 ProductName = "Chicken Delite",
 OrderId = "323B61",
 Price = 100,
 Category = "Non-Veg",
 Ingredients = "cheese, chicken chunks, onions & pineapple chunks.",
 ProductImage = ""
};
datas.Add(data);
data = new ProductList()
{
 ProductName = "Chicken Tikka",
 OrderId = "323B62",
 Price = 64,
 Category = "Non-Veg",
 Ingredients = "onions, grilled chicken, chicken salami & tomatoes.",
 ProductImage = ""
};
datas.Add(data);
return datas;
}
}
`
```

## Report parameters

Provides property options to pass or set report parameters default values at run-time using the `parameters` property. You can set the report parameters while creating the Report Viewer control in a application loaded.

In this tutorial, the `sales-order-dtail.rdl` report is used, and it can be downloaded from [here](#).

## Set report parameter

Set the default value data to the **Values** property and name of the report parameter to the **Name** property.

The parameter name is case sensitive, it should be same as available in the report definition.

The following code example illustrates how to set report parameter in the **ReportLoaded** event.

```
'csharp
this.ReportViewer.ReportLoaded += (sen, arg) =>
{
 List<BoldReports.UI.Xaml.ReportParameter> parameters = new
 List<BoldReports.UI.Xaml.ReportParameter>();
 BoldReports.UI.Xaml.ReportParameter parameter = new BoldReports.UI.Xaml.ReportParameter();
 parameter.Name = "SalesOrderNumber";
 parameter.Values = new List<string>() { "SO50756" };
 parameters.Add(parameter);
 this.ReportViewer.SetParameters(parameters);
};

this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportPath = "~/Resources/docs/sales-order-detail.rdl";
this.ReportViewer.RefreshReport();
'
```

## Set report parameter in Web API

Set the report path **ReportPath** and **ReportServiceURL** properties of the Report Viewer as in following code snippet.

```
'csharp
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportPath = "~/Resources/docs/sales-order-dtail.rdl";
this.ReportViewer.RefreshReport();
'
```

To set parameter default value in Web API Controller, use the following code in the **OnReportLoaded** method.

```
'csharp
```

```
public void OnReportLoaded(ReportViewerOptions reportOption)
{
 List<BoldReports.Web.ReportParameter> userParameters = new
 List<BoldReports.Web.ReportParameter>();
 userParameters.Add(new BoldReports.Web.ReportParameter()
 {
 Name = "SalesOrderNumber",
 Values = new List<string>() { "SO50756" }
 });
 reportOption.ReportModel.Parameters = userParameters;
}
```

### Get report parameter

Methods | Description

GetParameters | Returns the parameters that are used in the current report without the processed values.

GetParametersWithValues | Returns the report parameters with processed data values that are used in the current report.

You can use the following code sample to get parameter names and set parameter default values.

Set the report path `ReportPath` and `ReportServiceURL` properties of the Report Viewer as in following code snippet.

```
'csharp
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportPath = "~/Resources/docs/sales-order-dtail.rdl";
this.ReportViewer.RefreshReport();
'
```

To set parameter default value based on the parameter name in Web API Controller, use the following code in the `OnReportLoaded` method.

```
'csharp
public class ReportViewerController : ApiController, IReportController
{
 Dictionary<string, object> jsonArray = null;
 public object PostReportAction(Dictionary<string, object> jsonResult)
```

```
{
jsonArray = jsonResult;
return ReportHelper.ProcessReport(jsonResult, this);
}
....
public void OnReportLoaded(ReportViewerOptions reportOption)
{
var reportParameters = ReportHelper.GetParameters(jsonArray, this);
List<BoldReports.Web.ReportParameter> setParameters = new
List<BoldReports.Web.ReportParameter>();
if (reportParameters != null)
{
foreach (var rptParameter in reportParameters)
{
setParameters.Add(new BoldReports.Web.ReportParameter()
{
Name = rptParameter.Name,
Values = new List<string>() { "SO50756" }
});
}
reportOption.ReportModel.Parameters = setParameters;
}
}
}
`
```

## Report interaction events

You can handle the report processing actions and interact with reports using the following events.

[ReportLoaded](#)

[ReportError](#)

[Drill through](#)

## Report loaded

The **ReportLoaded** event fires once the report loading is completed and ready to start the processing. You can handle the event and specify data source, parameters at client-side. The following sample code loads a report by assigning report data source input in the **ReportLoaded** event.

In this tutorial, **product-list.rdlc** report is used, you can add the report from Bold Reports installation location. For more information, see [Samples and demos](#).

`'csharp`

```
Assembly assembly = typeof(MainPage).GetTypeInfo().Assembly;
Stream reportStream = assembly.GetManifestResourceStream("ReportViewer_UWP.Resources.product-
list.rdlc");
this.ReportViewer.ReportLoaded += (sen, arg) =>
{
 this.ReportViewer.DataSources.Clear();
 this.ReportViewer.DataSources.Add(new BoldReports.UI.Xaml.ReportDataSource { Name = "list", Value
= ProductList.GetData() });
};
this.ReportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
this.ReportViewer.LoadReport(reportStream);
this.ReportViewer.RefreshReport();
`
```

## Report error

When an error occurs in the report processing, it raises the **ReportError** event. You can handle the event and show the details in your custom dialog instead of component default error detail interface.

`'csharp`

```
this.ReportViewer.ProcessingMode = BoldReports.UI.Xaml.ProcessingMode.Local;
Assembly assembly = typeof(MainPage).GetTypeInfo().Assembly;
Stream reportStream = assembly.GetManifestResourceStream("ReportViewer_UWP.Resources.product-
list.rdlc");
this.ReportViewer.ReportError += (sen, arg) =>
{
 //Handle your report error options here.
}
//this.ReportViewer.LoadReport(reportStream);
this.ReportViewer.RefreshReport();
`
```

## Drill through

When a drill through item is selected in a report, it invokes the **DrillThroughReport** event. You can change the drill through arguments such as report parameter and data sources. The following sample code can be used to change the drill through report name and set the parameter value before the drill through report is rendered.

```
'csharp
this.ReportViewer.DrillThroughReport += (sen, arg) =>
{
 List<BoldReports.UI.Xaml.ReportParameter> parameters = new
 List<BoldReports.UI.Xaml.ReportParameter>();
 BoldReports.UI.Xaml.ReportParameter parameter = new BoldReports.UI.Xaml.ReportParameter();
 parameter.Name = "EmployeeID";
 parameter.Values = new List<string>() { "4" };
 parameters.Add(parameter);
 //Assign the drill through report path and parameter
 arg.ReportPath = "~/Resources/docs/personal-details.rdl";
 arg.Parameters = parameters;
};

this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportPath = "~/Resources/docs/sales-person-details.rdl";
this.ReportViewer.RefreshReport();
'
```

## Error logging in UWP Report Viewer

If an error occurred in report processing, UWP Report Viewer displays short messages about the error in view. In report viewer **Reporterror** event raised. You can register the event and get the report error details from the event arguments to save all logs and error information into a physical file location.

This section explains how to log the detailed error information to your UWP application.

This section requires a UWP Report Viewer application, if you don't have then create using the [Getting-Started](#).

In Solution Explorer, Open report viewer initialization cs file.

Register the **ReportError** event.

```
'csharp
private void MainPage_Loaded(object sender, RoutedEventArgs e)
{
```

```
this.ReportViewer.ReportError += ReportViewer_ReportError;
}

private void ReportViewer_ReportError(object sender, ReportErrorEventArgs e)
{
 // You can register the report errors using event arguments.
}
```

Create a method in `MainPage.xaml.cs` to write the error text into application folder.

```
'csharp
private async void WriteLogs(string errorMessage)
{
 var appFolder = Windows.Storage.ApplicationData.Current.LocalFolder;
 //you can refer to your preferred location path for errordetails text file.
 var filePath = await appFolder.CreateFileAsync("Errordetails.txt",
 Windows.Storage.CreationCollisionOption.OpenIfExists);
 await Windows.Storage.FileIO.AppendTextAsync(filePath, errorMessage + Environment.NewLine);
}
```

Invoke the newly created function in `ReportViewer_ReportError` as follows.

```
'csharp
private void ReportViewer_ReportError(object sender, ReportErrorEventArgs e)
{
 string errorLog;
 if (e.Exception != null)
 {
 errorLog = string.Format("Error Message: {0} \n Stack Trace: {1}", e.Message, e.Exception.StackTrace);
 }
 else
 {
 errorLog = e.Message;
 }
 WriteLogs(errorLog);
}
```

```
}
```

In cases of any issues faced in the report rendering, share the log file to our technical support team to get assistance on that.

The final `MainPage.xaml.cs` file is given as follows, you can replace it in your application.

```
'csharp
public sealed partial class MainPage : Page
{
 public MainPage()
 {
 this.InitializeComponent();
 this.Loaded += MainPage_Loaded;
 }

 private void MainPage_Loaded(object sender, RoutedEventArgs e)
 {
 this.ReportViewer.ReportPath = "~/Resources/docs/sales-order-dtail.rdl";
 this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
 this.ReportViewer.ReportError += ReportViewer_ReportError;
 this.ReportViewer.RefreshReport();
 }

 private void ReportViewer_ReportError(object sender, ReportErrorEventArgs e)
 {
 string errorLog;
 if (e.Exception != null)
 {
 errorLog = string.Format("Error Message: {0} \n Stack Trace: {1}", e.Message, e.Exception.StackTrace);
 }
 else
 {
 errorLog = e.Message;
 }
 WriteLogs(errorLog);
 }
}
```

Print report

Remove empty spaces in printing

```
private async void WriteLogs(string errorMessage)
{
 var appFolder = Windows.Storage.ApplicationData.Current.LocalFolder;
 //you can refer to your preferred location path for errordetails text file.
 var filePath = await appFolder.CreateFileAsync("Errordetails.txt",
 Windows.Storage.CreationCollisionOption.OpenIfExists);
 await Windows.Storage.FileIO.AppendTextAsync(filePath, errorMessage + Environment.NewLine);
}
```

## Print report

The Report Viewer provides print report option in the toolbar to print a copy of the report. The page setup dialog allows you to set the paper size or other page setup properties. To see print margins, click Print Layout on the toolbar.

The values allow you to set in the Page Setup dialog box for current session only. When you close the report and reopen it, it will have the default values again. The default values for the page setup dialog come from the report properties, which are set in the design view.

## Remove empty spaces in printing

The extra blank page is created when the Body of your report is too wide for your page. If you want the report to appear on a single page, all the content within the report body must fit on the physical page and the body width should be lesser or equal to the following formula:

Body Width <= Page Width - (Left Margin + Right Margin)

For more details on designing a report to remove the empty pages in the report, refer to the knowledge base article of [report page sizing](#).

## Export report

The Report Viewer provides events and properties to control and customize the report exporting functionality.

### PDF export options

The **PDFOptions** provides properties to manage PDF export behaviors. You have to set the properties in the application window loaded method.

### Export with complex scripts

To export reports with the complex scripts, set the **EnableComplexScript** property of **PDFOptions** instance to true.

'csharp

```
this.ReportViewer.PDFOptions = new BoldReports.Writer.PDFOptions();
this.ReportViewer.PDFOptions.EnableComplexScript = true;
```

## PDF Conformance

You can export the report as PDF/A-1b document by specifying the conformance level `PdfConformanceLevel.Pdf_A1B` in the `PdfConformanceLevel` property.

```
'csharp
```

```
this.ReportViewer.PDFOptions = new BoldReports.Writer.PDFOptions();
this.ReportViewer.PDFOptions.PdfConformanceLevel = Syncfusion.Pdf.PdfConformanceLevel.Pdf_A1B;
```

## Add custom PDF fonts

This allows you to have custom fonts in the PDF exported document by adding the font streams to `Fonts` collection in `PDFOptions` instance.

Add the font .ttf files into your application `Resources` folder.

In the Solution Explorer, open the properties of the font file and set the property `Copy to Output Directory` as `Copy always`.

Initialize the `Font` collection and add the font stream to it.

The key value provided in the font collection should be same as in the report item font property.

```
'csharp
```

```
this.ReportViewer.PDFOptions = new BoldReports.Writer.PDFOptions()
{
 Fonts = new Dictionary<string, System.IO.Stream>
 {
 { "Segoe UI", assembly.GetManifestResourceStream("ReportViewerUWP.Resources.fontsymbols.ttf") }
 }
};
```

Any fonts used in the report definition that is not installed or available in the local system, then you must load the font stream. In the above code, loaded `font_symbols` font stream to export `sales-order-detail.rdl` report as symbols.

## Word export options

The `WordOptions` provides properties to manage Word document export behaviors.

### Word document type

You can save the report to the required document version by setting the `FormatType` property.

```
'csharp
```

```
this.ReportViewer.WordOptions = new BoldReports.Writer.WordOptions();
this.ReportViewer.WordOptions.FormatType = BoldReports.Writer.WordFormatType.Docx;
`
```

### Word document advance layout for merged cells

Eliminate the tiny columns, rows, merged cells, and render the word document elements without nested grid layout by setting the **LayoutOption** as **TopLevel**. The **ParagraphSpacing** is the distance value added between two elements in the document.

```
`csharp
```

```
this.ReportViewer.WordOptions = new BoldReports.Writer.WordOptions();
this.ReportViewer.WordOptions.LayoutOption = BoldReports.Writer.WordLayoutOptions.TopLevel;
this.ReportViewer.WordOptions.ParagraphSpacing = new BoldReports.Writer.ParagraphSpacing()
{
 Bottom = 0.5f,
 Top = 0.5f
};
```

A paragraph element is inserted between two tables in the exported document to overcome word document auto merging behavior.

The table in word document is not a stand-alone object, if you draw two tables one after another, it will automatically get merged into a single table. To prevent this merging, added an empty paragraph between two tables.

### Protecting Word document from editing

You can restrict a Word document from editing either by providing a password or without a password. The following are the types of protection.

**AllowOnlyComments:** You can add or modify only the comments in the Word document.

**AllowOnlyFormFields:** You can modify the form field values in the Word document.

**AllowOnlyRevisions:** You can accept or reject the revisions in the Word document.

**AllowOnlyReading:** You can only view the content in the Word document.

**NoProtection:** You can access or edit the Word document contents as normally.

```
`csharp
```

```
this.ReportViewer.WordOptions = new BoldReports.Writer.WordOptions();
this.ReportViewer.WordOptions.ProtectionType = Syncfusion.DocIO.ProtectionType.AllowOnlyReading;
`
```

## Excel export options

The **ExcelOptions** provides properties to manage Excel document export behaviors.

### Excel document type

You can save the report to the required excel version by setting the **ExcelSaveType** property.

```
'csharp
```

```
this.ReportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions();
```

```
this.ReportViewer.ExcelOptions.ExcelSaveType = BoldReports.Writer.ExcelVersion.Excel2013;
```

```
'
```

### Excel document advance layout for merged cells

Eliminate the tiny columns, rows, and merged cells to provide clear readability and perform data manipulations by setting the **LayoutOption** as **IgnoreCellMerge**.

```
'csharp
```

```
this.ReportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions();
```

```
this.ReportViewer.ExcelOptions.LayoutOption =
BoldReports.Writer.ExcelLayoutOptions.IgnoreCellMerge;
```

```
'
```

### Protecting Excel document from editing

You can restrict the Excel document from editing either by providing the **ExcelSheetProtection** or enabling the **ReadOnlyRecommended** properties.

```
'csharp
```

```
this.ReportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions()
```

```
{
```

```
 ReadOnlyRecommended = true,
```

```
 ExcelSheetProtection = Syncfusion.XlsIO.ExcelSheetProtection.DeletingColumns
```

```
};
```

```
'
```

## PowerPoint export options

You can save the report to the required PowerPoint version by setting the **FormatType** property.

```
'csharp
```

```
this.ReportViewer.PPTOptions = new BoldReports.Writer.PPTOptions();
```

```
this.ReportViewer.PPTOptions.FormatType = BoldReports.Writer.PPTSaveType.PowerPoint2013;
```

```
'
```

## CSV export options

The **CsvOptions** allows you to change encoding, delimiters, qualifiers, extension, and line break of a CSV exported document.

```
'csharp
this.ReportViewer.CsvOptions = new BoldReports.Writer.CsvOptions()
{
 Encoding = System.Text.Encoding.Default,
 FieldDelimiter = ",",
 UseFormattedValues = false,
 Qualifier = "#",
 RecordDelimiter = "@",
 SuppressLineBreaks = true,
 FileExtension = ".txt"
};
```

### HTML export options

You can hide the separator added at the end of each page by setting the `HidePageSeparator` property to true.

```
'csharp
this.ReportViewer.HTMLOptions = new BoldReports.Writer.HTMLOptions();
this.ReportViewer.HTMLOptions.HidePageSeparator = true;
```

### Password protect exported document

This allows you to protect the exported document such as PDF, Word, Excel, and PowerPoint from unauthorized users by encrypting the document using encryption password. The following code snippet illustrates how to encrypt the exported document with the user defined password.

```
'csharp
//PDF encryption
this.ReportViewer.PDFOptions = new BoldReports.Writer.PDFOptions();
this.ReportViewer.PDFOptions.Security = new Syncfusion.Pdf.Security.PdfSecurity()
{
 UserPassword = "Password"
};

//Word encryption
this.ReportViewer.WordOptions = new BoldReports.Writer.WordOptions()
{
 EncryptionPassword = "password"
```

```
Handle post actions RenderingBegin

};

//Excel encryption

this.ReportViewer.ExcelOptions = new BoldReports.Writer.ExcelOptions()

{

 PasswordToModify = "password",

 PasswordToOpen = "password"

};

//PPT encryption

this.ReportViewer.PPTOptions = new BoldReports.Writer.PPTOptions()

{

 EncryptionPassword = "password"

};

`

Password protection is not supported for HTML export format.
```

## Handle post actions

You can handle the report processing post actions in following Report Viewer events to customize the rendered report.

## RenderingBegin

## RenderingCompleted

## ReportServiceRequestBegin

## EncryptCredentials

## RenderingBegin

This event occurs when the report begins rendering. Information about this event is passed in a `CancelEventArgs` object to a `CancelEventHandler` delegate, which handles the event. You can use the following code in your application.

`csharp

```
this.ReportViewer.RenderingBegin += async (sen, arg) =>
```

{

```
var dialog = new MessageDialog("This event handler will be called before the report rendering.");
await dialog.ShowAsync();
};
```

### RenderingCompleted

This event occurs when the report finishes rendering. Information about this event is passed in a `RenderingCompletedEventArgs` object to the `RenderingCompletedEventHandler` delegate, which handles the event. You can use the following code in your application.

```
'csharp
```

```
this.ReportViewer.RenderingCompleted += async (sen, arg) =>
{
 var dialog = new MessageDialog("This event handler will be called after complete the report
rendering.");
 await dialog.ShowAsync();
};
```

### ReportServiceRequestBegin

This event occurs when service request started for RDL processing mode to interact with web api for ReportViewer. You can use the following code in your application.

```
'csharp
```

```
this.ReportViewer.ReportServiceRequestBegin += async (sen, arg) =>
{
 var dialog = new MessageDialog("This event handler will be called before interact the report viewer web
api services");
 await dialog.ShowAsync();
};
```

### EncryptCredentials

Encrypt the report data source credentials and report server credential to pass the report viewer controller to render the report securely. You can use the following code in your application.

```
'csharp
```

```
this.ReportViewer.EncryptCredentials += (sen, arg) =>
{
 DataSourceCredentials credential = new DataSourceCredentials();
 credential.Name = "AdventureWorks";
 credential.UserId = "ssrs1";
 credential.Password = "RDLReport1";
 // Set the Report server credential and Data source credential for the report server reports.
 arg.ReportServerCredential = new System.Net.NetworkCredential("ssrs", "RDLReport1");
 arg.DataSourceCredentials.Add(credential);
};
```

};

## Localization of Bold Report Viewer

Localization of Bold Report Viewer allows you to localize the static text such as tooltip, parameter block, and dialog text based on a specific culture. Refer the following steps to localize the Report Viewer based on the culture.

Add Resources file for the different cultures.

Assign the value to each culture using key.

Assign a Current UI Culture to the application.

Set Report Viewer properties.

### Add Resources file for the different cultures

Right-click the project and add the **Resources** folder in your application.

Right-click on the **Resources** folder and add the new folder then name it **asfr-FR**.

Right-click on the **fr-FR** folder and press **Ctrl+Shift+A** keys or select **Add > New Item** from the context menu.

In the Add New Item dialog, select **Resources** file and name it as  
**BoldReports.UWP.Resources.resw**.

![Adding a new resource file](/static/assets/uwp/report-viewer/images/localization/add-new-resource-file.png)

Click **Add**.

In case, the another culture is used in the application, then create another folder in name **[CultureInfo Code]** under the **Resources** folder. For example, **fr-FR** and the naming convention needs to be followed mandatorily.

### Assign the value to each culture using key

To assign **Values** in Resource, the resource file needs to be updated according to the following steps.

Open the **BoldReports.UWP.Resources.resw** file by double clicking it from Solutions Explorer.

Add the key, Name, and its corresponding localized value by editing its field. Modify the resource values based on the **fr-FR** culture as shown in the following image.

![Edit resource file](/static/assets/uwp/report-viewer/images/localization/edit-resource-file.png)

|             |                                                |
|-------------|------------------------------------------------|
| Limitations | Assign a Current UI Culture to the application |
|-------------|------------------------------------------------|

you can download the modified resource file from [here](#) and replace it in your application.

### Assign a Current UI Culture to the application

Mention the culture to be referred while initializing the application, so that application refer to the appropriate value provided in resource file.

Report Viewer application culture can be changed by setting the **PrimaryLanguageOverride** in the **MainPage()** constructor. In this application, **MainPage.xaml.cs** is the startup page and culture is assigned as like below.

```
'csharp
public MainPage()
{
 this.InitializeComponent();
 ApplicationLanguages.PrimaryLanguageOverride = "fr-FR";
}
'
```

### Set Report Viewer properties

Add the Report Viewer initialization code.

```
'csharp
this.ReportViewer.ReportServiceURL = @"https://demos.boldreports.com/services/api/ReportViewer";
this.ReportViewer.ReportPath = "~/Resources/docs/sales-order-dtail.rdl";
this.ReportViewer.RefreshReport();
'
```

In this tutorial, **sales-oreder-detail** report is used.

Now, run the application and the below output shows the toolbar items localized **fr-FR** culture.

![Localization output](/static/assets/uwp/report-viewer/images/localization/localization-output.png)

## Limitations

### RDL specification

The Report Viewer control does not support RDL specification for SQL Server 2000 and SQL Server 2005.

### Report layout

In the Tablix cell split layout process, the entire cell moves to the next page to display the complete cell items, when the table cell width value exceeds the page width.

## Expressions

The object function and VB function do not have complete support.

## SSRS

The SSRS Report Server does not provide options to get credential information of the report data source deployed on the server. If the report has any data source that uses credentials to connect with the database, then you should specify the data source credentials for each report data source to establish database connection.

## Samples and Demos

Browse and explore the ready-to-use RDL, RDLC reports, samples, and offline demos.

### Locally installed reports

You can obtain the sample RDL and RDLC files from the Bold Reports installed location

`%localappdata%\Bold Reports\ReportsSDK\Samples\Common\Data\ReportTemplate.`

### Offline demos

The offline samples are provided in the Bold Reporting Tools setup. For more details, refer to the [Bold Reporting Tools sample deployment](#).

## Migrate Report Viewer application

In our Bold Reports new assemblies are introduced for both client and server-side to resolve the compatibility problem between Essential Studio Report Viewer versions. It has changes in both Web API service and client-side scripts.

This section provides step-by-step instructions for migrating Report Viewer from Syncfusion Essential Studio release version to Bold Reports version of UWP Report Viewer application:

### Client-side migration

In the Solution Explorer, right-click the **References** and remove the following assembly references:

`Syncfusion.SfReportViewer.UWP`

Add the assembly from the Syncfusion NuGet package `BoldReports.UWP`. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for `BoldReports.UWP` NuGet package, and install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

### Control initialization

Open the `MainWindow.xaml` file and remove the following namespace in your XAML page.

`'csharp`

`xmlns:syncfusion="using:BoldReports.UI.Xaml"`

Add the following namespace in your XAML page.

```
'csharp
xmlns:syncfusion="using:BoldReports.UI.Xaml"
```

## Server-side migration

In the Solution Explorer, right-click the **References** and remove the **Syncfusion.EJ.ReportViewer** assembly reference.

Add the assembly from the Bold Reporting NuGet package **BoldReports.Web**. To add from NuGet, right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages**. Search for **BoldReports.Web** NuGet package, and then install in your application.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

## Web API Controller

The **IReportController** interface is moved to **BoldReports.Web.ReportViewer**. Open the Report Viewer Web API Controller file and remove the following using statement.

```
'csharp
using Syncfusion.EJ.ReportViewer;
```

Add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
```

Your application is successfully upgraded to the latest version of Report Viewer, and you can run the application with new assemblies.

## Report export configuration

Now, the **BoldReports.Web** can export the reports with data visualization components only using web components. It is mandatory to configure the web scripts in Report Viewer Web API controller for exporting data visualization components such as chart, gauge, and map that are used in report definition. To configure the scripts in Web API, refer to the following steps:

Open the Report Viewer Web API controller.

Configure the following scripts and styles in **OnInitReportOptions** on Web API controller:

**jquery-1.10.2.min.js**

**bold.reports.common.min.js**

**bold.reports.widgets.min.js**

**ej.chart.min.js** - Exports the chart item. Add this script only if your report contains the chart report item.

**ej2-base.min.js**, **ej2-data.min.js**, **ej2-pdf-export.min.js**, **ej2-svg-base.min.js**, **ej2-lineargauge.min.js** and **ej2-circulargauge.min.js** - Exports the gauge item. Add this script only if your report contains the gauge report item.

**ej.map.min.js** - Exports the map item. Add this script only if your report contains the map report item.

**bold.report-viewer.min.js**

Replace the following codes in Report Viewer Web API controller.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 var resourcesPath = System.Web.Hosting.HostingEnvironment.MapPath("~/Scripts");
 reportOption.ReportModel.ExportResourceOption.Scripts = new List<string>
 {
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.widgets.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/bold.reports.common.min.js",
 //Chart component script
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.chart.min.js",
 //Gauge component scripts
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-data.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-pdf-export.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/common/ej2-svg-base.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-lineargauge.min.js",
 "https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej2-circulargauge.min.js",
 }
}
```

```
//Map component script
"https://cdn.boldreports.com/2.2.32/scripts/data-visualization/ej.map.min.js",
//Report Viewer Script
"https://cdn.boldreports.com/2.2.32/scripts/bold.report-viewer.min.js",
};
reportOption.ReportModel.ExportResourceOption.DependentScripts = new List<string>
{
 "https://code.jquery.com/jquery-1.10.2.min.js"
};
}
`
```

The data visualization components will not export without the above script configurations.

## NuGet Packages for UWP

Refer to the following steps to configure Bold Reporting NuGet packages for UWP application.

### Configure NuGet feed URL

#### Online NuGet feed URL

The Bold Reporting NuGet packages are published in [Nuget.org](#). To configure the online packages, use the following steps:

Open Visual Studio application.

On the **Tools** menu, select **Options**.

Expand the **NuGet Package Manager** and select **Package Sources**.

Click the **Add** button, enter the following **Package Name** and **Package Source URL**, and then click **Update**.

**Name:** [NuGet.org](#)

**Source:** <https://api.nuget.org/v3/index.json>

![Online NuGet Configure](/static/assets/uwp/report-viewer/images/nuget-packages/reporting-online-nuget-packages-configuration.png)

### Installing NuGet packages

#### Install using NuGet Package Manager

The NuGet Package Manager can be used to search and install NuGet packages in Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution**. Alternatively, right-click the project/solution in Solution Explorer tab, and choose **Manage NuGet Packages**.

By default, the **NuGet.org** package is selected in the **Package source** drop-down. Select package source, search for the **BoldReports.UWP** package, and then click **Install** button.

#### Install using Package Manager Console

To install the Reporting component using the Package Manager Console as NuGet packages,

On the **Tools** menu, select **NuGet Package Manager**, and then click **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

#### install specified package in default project

```
Install-Package <Package Name>
```

#### install specified package in default project with specified package source

```
Install-Package <Package Name> -Source <Source Location>
```

#### install specified package in specified project

```
Install-Package <Package Name> - ProjectName <Project Name>
```

For example:

```
'cmd
```

#### install specified package in default project

```
Install-Package BoldReports.UWP
```

#### install specified package in default project with specified Package Source

```
Install-Package BoldReports.UWP –Source “https://api.nuget.org/v3/index.json”
```

#### install specified package in specified project

```
Install-Package BoldReports.UWP -ProjectName BoldReportsApplication
```

#### Upgrading NuGet packages

##### Upgrading using NuGet Package Manager

NuGet packages can be updated to their specific version or latest version available in the Visual Studio solution or project:

On the **Tools** menu, click **NuGet Package Manager | Manage NuGet Packages for Solution....**.  
Alternatively, right-click the project/solution in the Solution Explorer tab, and then choose **Manage NuGet Packages**.

Select the **Updates** tab to see the packages available for update from the desired package sources, select the required packages and specific version from the drop-down, and then click the **Update** button.

### Upgrading using Package Manager Console

To update the installed Bold Reporting NuGet packages using the Package Manager Console:

On the **Tools** menu, select **NuGet Package Manager**, and then select **Package Manager Console**.

Run the following NuGet installation commands:

```
'cmd
```

### Update specific NuGet package in default project

```
Update-Package <Package Name>
```

### Update all the packages in default project

```
Update-Package
```

### Update specified package in default project with specified package source

```
Update-Package <Package Name> -Source <Source Location>
```

### Update specified package in specified project

```
Update-Package <Package Name> - ProjectName <Project Name>
```

**For example:**

```
'cmd
```

### Update specified Bold Reporting NuGet package

```
Update-Package BoldReports.UWP
```

### Update specified package in default project with specified Package Source

```
Update-Package BoldReports.UWP –Source “https://api.nuget.org/v3/index.json”
```

### Update specified package in specified project

```
Update-Package BoldReports.UWP -ProjectName BoldReportsApplication
```

### Upgrading using NuGet CLI

Using the NuGet CLI, all the NuGet packages in the project can be updated to the available latest version:

Download the latest [NuGet CLI](#).

To update the existing `nuget.exe` to latest version use the following command:

```
'cmd
nuget update -self
'
```

Open the downloaded executable location in the command window. Run the following “update commands” to update the Bold Reporting NuGet packages.

```
'cmd
```

### update all NuGet packages from config file

`nuget update <configPath> [options]`

### update all NuGet packages from specified Packages Source

`nuget update -Source <Source Location> [optional]`

`configPath` is optional. It identifies the `package.config` or solutions file lists the packages utilized in the project.

**For example:**

```
'cmd
```

### Update all NuGet packages from config file

`nuget update "C:\Users\BoldReportsApplication\package.config"`

### Update all NuGet packages from specified Packages Source

`nuget update -Source "https://api.nuget.org/v3/index.json"`

The Update command is not working as expected in Mono (Mac and Linux) and projects using PackageReference format.

### Frequently asked questions

This section helps to get the answer for the frequently asked questions in Bold Reports UWP Report Viewer.

[How can improve the performance and handle the large amounts of data with Report Viewer?](#)

[Is Report Viewer compatible with latest version of JQuery library?](#)

[Is the back action from drillthrough report will load the report again with Report Viewer?](#)

## UWP Report Viewer Reporting Service

The UWP Report Viewer requires a Web API service to process the report files. The following topics explains how to create reporting Web API service to preview the reports.

[ASP.NET Web API Service](#)

[ASP.NET Core Web API Service](#)

## Create ASP.NET Web API service

In this section, you will learn how to create a Web API Service for Report Viewer using the new ASP.NET Empty Web Application template.

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select the required **.NET Framework** in the drop-down.

Select **ASP.NET Web Application (.NET Framework)**, change the application name, and then click **OK**.

![ASP.NET Web Application project template](/static/assets/uwp/report-viewer/images/report-service/aspnetmvc5-template.png)

Choose **Empty, Web API** and then click **OK**. Now, the Web application project is created with Web API.

![Select Web API and Empty options](/static/assets/uwp/report-viewer/images/report-service/add-web-api.png)

## Configure Report Viewer Web API

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, select the **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Refer to the [NuGet Packages](#) section to learn more details about installing and configuring Report Viewer NuGet packages.

Search for **BoldReports.Web** NuGet packages, and install them in your Web application.

Package | Purpose

PostReportAction | Action (HttpPost) method for posting the request in report process.

OnInitReportOptions | Report initialization method that occurs when the report is about to be processed.

OnReportLoaded | Report loaded method that occurs when the report and sub report start loading.

GetResource | Action (HttpGet) method to get resource for the report.

#### ReportHelper

The class **ReportHelper** contains helper methods that help to process a Post or Get request from the Report Viewer control and return the response to the Report Viewer control. It has the following methods:

Methods | Description

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

#### Add Web API Controller

Right-click **Controller** folder in your project and select **Add > New Item** from the context menu.

Select **Web API Controller Class** from the listed templates and name it as  
**ReportViewController.cs**

![Provide controller name](/static/assets/uwp/report-viewer/images/report-service/add-web-api-controller.png)

Click **Add**.

While adding Web API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and implement its methods (replace the following code in newly created Web API controller).

```
'csharp
public class ReportViewController : ApiController, IReportController
{
 // Post action for processing the RDL/RDLC report
 public object PostReportAction(Dictionary<string, object> jsonResult)
```

```
{
 return ReportHelper.ProcessReport(jsonResult, this);
}

// Get action for getting resources from the report
[System.Web.Http.ActionName("GetResource")]
[AcceptVerbs("GET")]

public object GetResource(string key, string resourcetype, bool isPrint)
{
 return ReportHelper.GetResource(key, resourcetype, isPrint);
}

// Method that will be called when initialize the report options before start processing the report
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 // You can update report options here
}

// Method that will be called when reported is loaded
public void OnReportLoaded(ReportViewerOptions reportOption)
{
 // You can update report options here
}
}
`
```

#### Add routing information

To configure routing to include an action name in the URI, open the `WebApiConfig.cs` file and change the `routeTemplate` in the `Register` method as follows,

```
'csharp
public static class WebApiConfig
{
 public static void Register(HttpConfiguration config)
 {
 // Web API configuration and services
 // Web API routes
```

```
config.MapHttpAttributeRoutes();
config.Routes.MapHttpRoute(
 name: "DefaultApi",
 routeTemplate: "api/{controller}/{action}/{id}",
 defaults: new { id = RouteParameter.Optional }
);
}
}
`
```

Compile and run the Web API service application.

#### Enable Cross-Origin Requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution** menu command.

Search for **Microsoft.AspNet.WebApi.Cors** NuGet packages, and install them in your Web API application.

Call **EnableCors** in **WebApiConfig** to add CORS services to the **Register** method. Replace the following code to allow any origin requests.

```
`csharp
public static class WebApiConfig
{
 public static void Register(HttpConfiguration config)
 {
 // Add Enable Cors
 config.EnableCors();

 // Web API configuration and services
 // Web API routes
 config.MapHttpAttributeRoutes();
 config.Routes.MapHttpRoute(
 name: "DefaultApi",
```

```
routeTemplate: "api/{controller}/{action}/{id}",
defaults: new { id = RouteParameter.Optional }
);
}
}
`
```

To specify the CORS policy for API controller, add the `[EnableCors]` attribute to the controller class. Specify the policy name.

```
`csharp
[EnableCors(origins: "", headers: "", methods: "*")]
public class ReportViewerController : ApiController, IReportController
{
 // Post action for processing the RDL/RDLC report
 public object PostReportAction(Dictionary<string, object> jsonResult)
 {
 return ReportHelper.ProcessReport(jsonResult, this);
 }

 // Get action for getting resources from the report
 [System.Web.Http.ActionName("GetResource")]
 [AcceptVerbs("GET")]
 public object GetResource(string key, string resourcetype, bool isPrint)
 {
 return ReportHelper.GetResource(key, resourcetype, isPrint);
 }

 // Method that will be called when initialize the report options before start processing the report
 public void OnInitReportOptions(ReportViewerOptions reportOption)
 {
 // You can update report options here
 }

 // Method that will be called when reported is loaded
 public void OnReportLoaded(ReportViewerOptions reportOption)
 {
 // You can update report options here
 }
}
```

```
}
```

```
}
```

```
'
```

## Create ASP.NET Core Web API Service

To create an ASP.NET Core Web API for Report Viewer using a new ASP.NET Core Web Application template, follow these steps:

Open Visual Studio 2017, click the **File** menu, go to **New**, and then select **Project**.

Go to **Installed > Visual C# > Web**, and then select **ASP.NET Core Web Application**, change the application name, and then click **OK**.

Choose the **ASP.NET Core version** and select the **API application** template and click **OK**. Do not select Enable Docker Support.

![Creating a new ASP.NET Core Application Project](/static/assets/uwp/report-viewer/images/report-service/aspnet-core-web-application-template.png)

## List of dependency Libraries

The Web API service configuration requires the following reporting server-side packages. Right-click the project or solution in the **Solution Explorer** tab, and choose **Manage NuGet Packages** and then search for **BoldReports.Net.Core** package, and install to the application. The following provides detail of the packages and its usage.

### Package | Purpose

**Syncfusion.Compression.Net.Core** | Supports for exporting the report to PDF, Microsoft Word, and Microsoft Excel format. It is a base library for the packages **Syncfusion.Pdf.Net.Core** , **Syncfusion.DocIO.Net.Core** and **Syncfusion.XlsIO.Net.Core**.

**Syncfusion.Pdf.Net.Core** | Supports for exporting the report to a PDF.

**Syncfusion.DocIO.Net.Core** | Supports for exporting the report to a Word.

**Syncfusion.XlsIO.Net.Core** | Supports for exporting the report to an Excel.

**Syncfusion.OfficeChart.Net.Core** | It is a base library of the **Syncfusion.XlsIO.Net.Core** package.

**Newtonsoft.Json** | Serialize and deserialize the data for report viewer. It is a mandatory package for the report viewer, and the package version should be higher of 10.0.1 for NET Core 2.0 and others should be higher of 9.0.1.

**System.Data.SqlClient** | This is an optional package for the report viewer. It should be referred in project when renders the RDL report and which contains the SQL Server and SQL Azure data source. Also, the package version should be higher of 4.1.0.

## Configure Web API

The interface **IReportController** has declaration of action methods that are defined in the Web API Controller for processing the RDL, RDLC, and SSRS reports and for handling request from the Report Viewer control. The **IReportController** has the following action methods declaration:

**Methods | Description**

GetResource | Returns the report resource to the requested key.

ProcessReport | Processes the report request and returns the result.

[Add Web API Controller](#)

Right-click the project and select **Add > New Item** from the context menu.

In the Add New Item dialog, select **API Controller** class and name it as **ReportViewerController.cs**

![Adding a new controller to the project](/static/assets/uwp/report-viewer/images/report-service/add-aspnet-core-api-controller.png)

Click **Add**.

While adding API Controller class, name it with the suffix **Controller** that is mandatory.

Open the **ReportViewerController** and add the following using statement.

```
'csharp
using BoldReports.Web.ReportViewer;
'
```

Inherit the **IReportController** interface, and then implement its methods.

Create local references for the interfaces given in following table.

**Interface | Purpose**

**IMemoryCache** | Report Viewer requires a memory cache to store the information of consecutive client request and have the rendered report viewer information in server.

**IHostingEnvironment** | IHostingEnvironment used to get the report stream from application **wwwroot\Resources** folder.

You cannot load the application report with path information in ASP.NET Core service. So, you should load the report as stream in **OnInitReportOptions**.

```
'csharp
public void OnInitReportOptions(ReportViewerOptions reportOption)
{
 string basePath = _hostingEnvironment.WebRootPath;
 // Here, we have loaded the sales-order-detail.rdl report from application the folder
 // wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
```

```
System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-detail.rdl", System.IO.FileMode.Open, System.IO.FileAccess.Read);

reportOption.ReportModel.Stream = reportStream;

}

`
```

You can replace the template code with the following code.

```
`csharp

[Route("api/[controller]/[action]")]
public class ReportViewerController : Controller, IReportController
{
 // Report viewer requires a memory cache to store the information of consecutive client request and
 // have the rendered report viewer information in server.

 private Microsoft.Extensions.Caching.Memory.IMemoryCache _cache;
 // IHostingEnvironment used with sample to get the application data from wwwroot.

 private Microsoft.AspNetCore.Hosting.IHostingEnvironment _hostingEnvironment;
 // Post action to process the report from server based json parameters and send the result back to the
 // client.

 public ReportViewerController(Microsoft.Extensions.Caching.Memory.IMemoryCache memoryCache,
 Microsoft.AspNetCore.Hosting.IHostingEnvironment hostingEnvironment)
 {
 _cache = memoryCache;
 _hostingEnvironment = hostingEnvironment;
 }

 // Post action to process the report from server based json parameters and send the result back to the
 // client.

 [HttpPost]
 public object PostReportAction([FromBody] Dictionary<string, object> jsonArray)
 {
 return ReportHelper.ProcessReport(jsonArray, this, this._cache);
 }

 // Method will be called to initialize the report information to load the report with ReportHelper for
 // processing.

 public void OnInitReportOptions(ReportViewerOptions reportOption)
 {
```

```
string basePath = _hostingEnvironment.WebRootPath;
// Here, we have loaded the sales-order-detail.rdl report from application the folder
// wwwroot\Resources. sales-order-detail.rdl should be there in wwwroot\Resources application folder.
System.IO.FileStream reportStream = new System.IO.FileStream(basePath + @"\Resources\sales-order-
detail.rdl", System.IO.FileMode.Open, System.IO.FileAccess.Read);
reportOption.ReportModel.Stream = reportStream;
}

// Method will be called when reported is loaded with internally to start to layout process with
ReportHelper.

public void OnReportLoaded(ReportViewerOptions reportOption)
{
}

//Get action for getting resources from the report
[ActionName("GetResource")]
[AcceptVerbs("GET")]

// Method will be called from Report Viewer client to get the image src for Image report item.
public object GetResource(ReportResource resource)
{
 return ReportHelper.GetResource(resource, this, _cache);
}

[HttpPost]
public object PostFormReportAction()
{
 return ReportHelper.ProcessReport(null, this, _cache);
}
`
```

The `sales-order-detail.rdl` report can be downloaded from [here](#). Also, you can add the report from Syncfusion installation location. For more information on installed sample location, see [Samples and demos](#).

Run the application and use the API URL in the Report Viewer `reportServiceUrl` property.

If you are using .NET Core 3.0 and above, refer to the [how to use the Bold Reports Embedded Reporting Tools with ASP.NET Core 3.x](#) section.

### Enable Cross-Origin requests

Browser security prevents Report Viewer from making requests to your Web API Service when both runs in a different domain. To allow access to your Web API service from a different domain, you must enable cross-origin requests.

Call `AddCors` in `Startup.ConfigureServices` to add the CORS services to the app's service container. Replace the following code to allow any origin requests.

```
'csharp
public void ConfigureServices(IServiceCollection services)
{
 services.AddMvc();
 services.AddCors(o => o.AddPolicy("AllowAllOrigins", builder =>
 {
 builder.AllowAnyOrigin()
 .AllowAnyMethod()
 .AllowAnyHeader();
 }));
}
```

To specify the CORS policy for the controller, add the `[EnableCors]` attribute to the controller class. Specify the policy name.

```
'csharp
[Microsoft.AspNetCore.Cors.EnableCors("AllowAllOrigins")]
public class ReportViewerController : Controller, IReportController
{
 public IActionResult Index()
 {
 return View();
 }

}
```

## How to Create RDL/RDLC Report

The following sections explain about how to create a new RDL/RDLC report using Bold Report Designer, Microsoft Report Builder and Visual Studio Report Server project template.

[Create a RDL report](#)

[Create a RDLC report](#)

## Create a SSRS RDL report

You can create an RDL report using any of the following reporting tools:

Bold Reports Report Designer.

Microsoft Report Builder.

Visual Studio Report Server project template.

### Bold Reports Report Designer

Bold Reports Report Designer provides the intuitive user interface to create and edit the RDL reports, which is available in Bold Embedded Reporting Tools Control Panel Add On.

![Add on for Report Designer](/static/assets/javascript/report-viewer/images/faq/add-on-for-report-designer/add-on-report-designer.png)

### Microsoft SQL Report Builder

You can create an RDL report using the Microsoft stand-alone Report Builder. For more details, refer to this [online documentation](#).

### Visual Studio Report Server template

To create an RDL report in Visual Studio, a Report Server project is required where you can save your report definition (.rdl) file. For more details, refer to this [Visual Studio documentation](#).

If you do not have the Business Intelligence or Report Server Project options, you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

## Create a RDLC report using business object data source

This section describes step by step procedure to create an RDLC report using Visual Studio Reporting project type.

### Prerequisites

Microsoft Visual Studio 2017 or higher

[Microsoft RDLC Report Designer](#)

If you are using Microsoft Visual Studio lower to 2017 version then you should update SSDT with the Business Intelligence templates. Refer to [Download SQL Server Data Tools](#).

### Create business object class

Open Visual Studio from the File menu and select **New Project**.

Create project with class library type from the project type list

![Add a new class library project](/static/assets/uwp/report-viewer/images/how-to/create-report/class-library-project.png)

Create the class with necessary properties. You can find the reference below,

```
'csharp
public class ProductSales
{
 public string ProdCat { get; set; }
 public string SubCat { get; set; }
 public string OrderYear { get; set; }
 public string OrderQtr { get; set; }
 public double Sales { get; set; }
}
```

Clean and build the application.

## Add an RDLC report

Right-click the project and click **Add > New Item**.

Search Report with new item and select **Report Wizard** to start the report creation with dataset selection.

Click **Add**.

![Add a new rdlc using report wizard template](/static/assets/uwp/report-viewer/images/how-to/create-report/add-sales-report-rdlc.png)

## Data source and table configuration wizard

Choose object type from the Data Source Configuration wizard and click **Next**.

![Select data source type in configuration wizard](/static/assets/uwp/report-viewer/images/how-to/create-report/choose-data-source-type.png)

Expand the tree view and select **ProductSales**, and then click **Finish**.

![Choose data object class in the application namespace](/static/assets/uwp/report-viewer/images/how-to/create-report/select-data-objects.png)

In the DataSet Properties wizard, specify the dataset name as **SalesData**.

![Set rdlc dataset properties](/static/assets/uwp/report-viewer/images/how-to/create-report/rdlc-dataset-properties.png)

Drag the fields into Values, Row, and Column groups, and then click **Next**.

![Arrange table row, column and value groups](/static/assets/uwp/report-viewer/images/how-to/create-report/arrange-table-fields.png)

Choose the table layout and click **Next**.

Select table style and click **Finish**.

![Choose table toggle, repeat header and total options](/static/assets/uwp/report-viewer/images/how-to/create-report/choose-table-layout.png)

Now, the RDLC report is displayed in the Visual Studio as follows.

![Visual Studio design output of the sales report](/static/assets/uwp/report-viewer/images/how-to/create-report/sales-report-design.png)

## Bold Report Writer

Report Writer is a class library that enables the user to render reports defined in Microsoft's RDL format (2008 or 2008 R2) as PDF, Word, Excel or CSV documents.

The important features of UWP Report Writer are listed as follows:

RDL Specification - Supports RDL specification for the SQL Server 2008 and RDL specification for the SQL Server 2008 R2 only. List of available report definition formats:  
[msdn.microsoft.com/library/dd297486\(SQL.100\)](http://msdn.microsoft.com/library/dd297486(SQL.100)).

Data sources - You can use advanced database servers Data Sources in Report Writer (SQL and Oracle).

Charts - Show all basic types of charts that are available in Microsoft RDL reports.

Tablix - Shows the summaries and simple tables.

Gauge - Shows measurement values by using expression values.

Textbox - Shows textbox data with expression support.

Export - Export report as PDF, Word, Excel and CSV.

Report Parameter - Views the report based on the report parameter value.

## Export SSRS RDL Report in Bold Reports UWP Report Writer

The Report Writer is a class library that is used to export the RDL report with popular file formats like PDF, Microsoft Word, Microsoft CSV, and Microsoft Excel without previewing the report in webpage. This section describes how to export the RDL report in UWP application using the [Report Writer](#).

### Create UWP application

Start Visual Studio 2019 and click **Create new project**.

Choose **Blank App (Universal Windows)**, and then click **Next**.

![UWP application project template](/static/assets/uwp/report-writer/images/getting-started/new-uwp-application.png)

Change the project name, and then click **Create**.

Select the target and minimum platform version **Windows 10, version 1809(10.0; Build 17763)** from the dropdown, and then click **OK**.

![New universal windows platform project](/static/assets/uwp/report-writer/images/getting-started/new-universal-windows-platform-project.png)

Set same version for the target and minimum platform version.

### Configure Report Writer in an application

Right-click the project or solution in the Solution Explorer tab, and choose **Manage NuGet Packages**.

Alternatively, go to **Tools > NuGet Package Manager > Manage NuGet Packages for Solution**.

Refer to the [NuGet Packages](#) to learn more details about installing and configuring Report writer NuGet packages. You can add the reports from Syncfusion installation location. For more information, refer to the [samples and demos](#) section.

Search for **BoldReports.UWP** NuGet package, and install them in your UWP application.

#### Package | Purpose

**BoldReports.UWP** | Contains UWP Reporting controls (Report Writer) to preview and export the reports.

#### Adding already created report

In this tutorial, the **Product List.rdlc** report is used, and it can be downloaded in this [link](#).

Create a folder **Resources** in your application to store the RDLC reports.

Add already created reports to the newly created folder.

## Initialize Report Writer

Initialize Report Writer export type inside the `tag` as shown below in `MainWindow.xaml` file,

```
'csharp
<Page
.....
.....
.....
.....
xmlns:BoldReports="using:BoldReports.UI.Xaml"
....>
<Grid Margin="0" Name="grd_controlPanel" Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
<StackPanel Name="pagePanel" Grid.Row="1" Margin="20,40,0,0" Background="White">
<TextBlock FontSize="15" FontFamily="Segoe UI Regular" TextWrapping="Wrap" Padding="5,5,5,5" >
<TextBlock.Text>This sample shows the capability of exporting a RDLC report into various file formats like PDF, WORD, EXCEL and HTML using Local export mode of Report Writer. Choose a file format and click generate button to view the selected document generated from report file.</TextBlock.Text>
</TextBlock>
<StackPanel Margin="0,10,0,0" Orientation="Vertical">
<RadioButton FontSize="15" FontFamily="Segoe UI Regular" Content="PDF" HorizontalAlignment="Left" Margin="20,10,0,0" x:Name="pdf" IsChecked="true" VerticalAlignment="Top"/>
<RadioButton FontSize="15" FontFamily="Segoe UI Regular" Content="Word" HorizontalAlignment="Left" Margin="20,10,0,0" x:Name="word" VerticalAlignment="Top"/>
<RadioButton FontSize="15" FontFamily="Segoe UI Regular" Content="Excel" HorizontalAlignment="Left" Margin="20,10,0,0" x:Name="excel" VerticalAlignment="Top"/>
<RadioButton FontSize="15" FontFamily="Segoe UI Regular" Content="HTML" x:Name="html" VerticalAlignment="Top" Margin="20,10,0,0" Width="90"/>
<Button Click="Button_Click" HorizontalAlignment="Left" Margin="20,10,0,0" VerticalAlignment="Bottom" BorderBrush="LightBlue" Background="#8bb54a">
<StackPanel Orientation= "Horizontal" Background="#8bb54a" Width="144">
<TextBlock Foreground="#ffffff" FontSize="16" FontFamily="Segoe UI Bold" Text="Generate" HorizontalAlignment="Right" Margin="30,5,0,0" Width="110" VerticalAlignment="Center" Height="27"/>
</StackPanel>
</Button>
</StackPanel>
```

```
</StackPanel>
</Grid>
</Page>
`
```

### [Set Report Path](#)

Open `MainWindow.xaml.cs` file and add the following using statement.

```
`csharp
using BoldReports.Writer;
using Windows.Storage.Pickers;
using System.Reflection;
using Windows.Storage.Streams;
using Windows.Storage;
using Windows.UI.Popups;
using System.Collections;
`
```

Create a class and methods that returns business object data collection. Use the following code in your application.

```
`csharp
public class ReportData
{
 public string ProdCat { get; set; }
 public string SubCat { get; set; }
 public double? OrderYear { get; set; }
 public string OrderQtr { get; set; }
 public double? Sales { get; set; }
 public static IList GetData()
 {
 List<ReportData> datas = new List<ReportData>();
 ReportData data = null;
 data = new ReportData()
 {
 ProdCat = "Accessories",
`
```

```
SubCat = "Helmets",
OrderYear = 2002,
OrderQtr = "Q1",
Sales = 4945.6925
};

datas.Add(data);
data = new ReportData()
{
 ProdCat = "Components",
 SubCat = "Road Frames",
 OrderYear = 2002,
 OrderQtr = "Q3",
 Sales = 957715.1942
};

datas.Add(data);
data = new ReportData()
{
 ProdCat = "Components",
 SubCat = "Forks",
 OrderYear = 2002,
 OrderQtr = "Q4",
 Sales = 23543.1060
};

datas.Add(data);
data = new ReportData()
{
 ProdCat = "Bikes",
 SubCat = "Road Bikes",
 OrderYear = 2002,
 OrderQtr = "Q1",
 Sales = 3171787.6112
};

datas.Add(data);
```

```
data = new ReportData()
{
 ProdCat = "Accessories",
 SubCat = "Helmets",
 OrderYear = 2002,
 OrderQtr = "Q3",
 Sales = 33853.1033
};

datas.Add(data);

return datas;
}

}
```

Initialize ReportWriter by using the following code example in the `MainWindow.xaml.cs` file export button click event.

```
'csharp
async void Button_Click(object sender, RoutedEventArgs e)
{
 FileSavePicker fileSavePicker = new FileSavePicker();
 WriterFormat format = WriterFormat.PDF;
 if (pdf.IsChecked == true)
 {
 fileSavePicker.FileTypeChoices.Add("PDF", new List<string> { ".pdf" });
 fileSavePicker.DefaultFileExtension = ".pdf";
 format = WriterFormat.PDF;
 }
 else if (excel.IsChecked == true)
 {
 fileSavePicker.FileTypeChoices.Add("Excel", new List<string> { ".xls" });
 fileSavePicker.DefaultFileExtension = ".xls";
 format = WriterFormat.Excel;
 }
 else if (word.IsChecked == true)
```

```
{
 fileSavePicker.FileTypeChoices.Add("Word", new List<string> { ".doc" });
 fileSavePicker.DefaultFileExtension = ".doc";
 format = WriterFormat.Word;
}

else if (html.IsChecked == true)
{
 fileSavePicker.FileTypeChoices.Add("Html", new List<string> { ".html" });
 fileSavePicker.DefaultFileExtension = ".html";
 format = WriterFormat.HTML;
}

fileSavePicker.SuggestedFileName = "ExportReport";
var savedItem = await fileSavePicker.PickSaveFileAsync();
if (savedItem != null)
{
 MemoryStream exportFileStream = new MemoryStream();
 Assembly assembly = typeof(MainPage).GetTypeInfo().Assembly;
 // Ensure the report location and application name.
 Stream reportStream = assembly.GetManifestResourceStream("<application name>.Resources.Product
List.rdlc");
 BoldReports.UI.Xaml.ReportDataSourceCollection datas = new
 BoldReports.UI.Xaml.ReportDataSourceCollection();
 datas.Add(new BoldReports.UI.Xaml.ReportDataSource { Name = "Sales", Value = ReportData.GetData()
});
 ReportWriter writer = new ReportWriter(reportStream, datas);
 writer.ExportMode = ExportMode.Local;
 writer.ExportCompleted += Writer_ExportCompleted;
 await writer.SaveASync(exportFileStream, format);
 try
 {
 using (IRandomAccessStream stream = await savedItem.OpenAsync(FileAccessMode.ReadWrite))
 {
 // Write compressed data from memory to file
 using (Stream outstream = streamAsStreamForWrite())
```

```
{
byte[] buffer = exportFileStream.ToArray();
outstream.Write(buffer, 0, buffer.Length);
outstream.Flush();
}
}
exportFileStream.Dispose();
}
}
catch {}
}
}
private void Writer_ExportCompleted(object sender, byte[] e)
{
MessageDialog msgDialog = new MessageDialog("Report exporting completed successfully");
msgDialog.ShowAsync();
}
`
```

Buils and run the application. Choose a file format and click generate button to view the selected document generated from report file.

Congratulations! you have completed your first UWP Writer application!.Click [here](#) to download the already created UWP Report Writer application.

## Embedded reporting tools licensing overview

Starting with the version 2.x, Syncfusion introduced a new licensing system for the Bold Reports. You have to include the license token in your projects in order to use the Bold Reports assemblies. Note that the license token is different from the setup unlock key and that needs to be generated separately from the [Bold Reports](#) website. The below licensing error will be displayed if the license token is missing in your projects.

This application was built using a trial version of Bold Reports. Please include a valid license to permanently remove this license validation message. You can also obtain a free 30 day evaluation license to temporarily remove this message during the evaluation period.

## How to generate Bold Reports license token

License tokens can be generated from the [Downloads](#) section of the [Bold Reports](#) site.

Bold Reports license tokens are version specific. So, you should use the corresponding version license tokens in the projects.

## How to register the Bold Reports license tokens

The generated license token is just a string that needs to be registered before any Bold Reports control is initiated. The following code is used to register the license.

'csharp

```
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
'
```

## ASP.NET

Register the license token in Application\_Start method of `Global.asax.cs/Global.asax`

'csharp

```
void Application_Start(object sender, EventArgs e)
{
 //Register Bold license
 Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
 // Code that runs on application startup
 RouteConfig.RegisterRoutes(RouteTable.Routes);
 BundleConfig.RegisterBundles(BundleTable.Bundles);
}
```

## ASP.NET MVC

Register the license token in Application\_Start method of `Global.asax.cs`

'csharp

```
void Application_Start(object sender, EventArgs e)
{
 //Register Bold license
 Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
 // Code that runs on application startup
 RouteConfig.RegisterRoutes(RouteTable.Routes);
 BundleConfig.RegisterBundles(BundleTable.Bundles);
}
```

## ASP.NET Core

Register the license token in `Configure` method of `Startup.cs`

'csharp

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env, ILoggerFactory loggerFactory)
```

```
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
loggerFactory.AddConsole(Configuration.GetSection("Logging"));
loggerFactory.AddDebug();
}
`
```

### WPF

Register the license token in App constructor of `App.xaml.cs` in C#. If App constructor not available in `App.xaml.cs`, create the `App()` constructor in `App.xaml.cs` and register the license token inside the constructor.

```
`csharp
public partial class App : Application
{
public App()
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
}
}
`
```

### UWP

Register the license token in `App.xaml.cs` constructor before `InitializeComponent()` in C#. If App constructor not available in `App.xaml.cs`, create the `App()` constructor in `App.xaml.cs` and register the license token inside the constructor.

```
`csharp
public App()
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
this.InitializeComponent();
this.Suspending += OnSuspending;
}
`
```

## Embedded reporting tools licensing overview

Starting with the version 2.x, Syncfusion introduced a new licensing system for the Bold Reports. You have to include the license token in your projects in order to use the Bold Reports assemblies. Note that the license token is different from the setup unlock key and that needs to be generated separately from the [Bold Reports](#) website. The below licensing error will be displayed if the license token is missing in your projects.

This application was built using a trial version of Bold Reports. Please include a valid license to permanently remove this license validation message. You can also obtain a free 30 day evaluation license to temporarily remove this message during the evaluation period.

### How to generate Bold Reports license token

License tokens can be generated from the [Downloads](#) section of the [Bold Reports](#) site.

Bold Reports license tokens are version specific. So, you should use the corresponding version license tokens in the projects.

### How to register the Bold Reports license token

The generated license tokens is just a string that needs to be registered before any Bold Reports control is initiated. The following code is used to register the license.

'csharp

```
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
'
```

### ASP.NET

Register the license token in Application\_Start method of [Global.asax.cs/Global.asax](#)

'csharp

```
void Application_Start(object sender, EventArgs e)
{
 //Register Bold license

 Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
 // Code that runs on application startup

 RouteConfig.RegisterRoutes(RouteTable.Routes);
 BundleConfig.RegisterBundles(BundleTable.Bundles);
}
```

### ASP.NET MVC

Register the license token in Application\_Start method of [Global.asax.cs](#)

'csharp

```
void Application_Start(object sender, EventArgs e)
{
```

```
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
// Code that runs on application startup
RouteConfig.RegisterRoutes(RouteTable.Routes);
BundleConfig.RegisterBundles(BundleTable.Bundles);
}
```

#### ASP.NET Core

Register the license token in `Configure` method of `Startup.cs`

```
'csharp
public void Configure(IApplicationBuilder app, IHostingEnvironment env, ILoggerFactory loggerFactory)
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
loggerFactory.AddConsole(Configuration.GetSection("Logging"));
loggerFactory.AddDebug();
}
```

#### WPF

Register the license token in App constructor of `App.xaml.cs` in C#. If App constructor not available in `App.xaml.cs`, create the `App()` constructor in `App.xaml.cs` and register the license token inside the constructor.

```
'csharp
public partial class App : Application
{
public App()
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
}
}
```

## UWP

Register the license token in App.xaml.cs constructor before InitializeComponent() in C#. If App constructor not available in App.xaml.cs, create the App() constructor in App.xaml.cs and register the license token inside the constructor.

'csharp

```
public App()
{
 //Register Bold license
 Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE TOKEN");
 this.InitializeComponent();
 this.Suspending += OnSuspending;
}
```

## Embedded reporting tools license token errors

Licensing error popup is displayed with various messages under different circumstances. Here are some ways to resolve different issues.

### License token not registered

The following error message will be shown if Bold license token was not registered in your application.

#### Error message:

This application was built using a trial version of Bold Reports. Please include a valid license to permanently remove this license validation message. You can also obtain a free 30 day evaluation license to temporarily remove this message during the evaluation period. Please refer to this [help topic](https://documentation.boldreports.com/licensing/license-token/) for more information.

#### Solution:

Generate a valid license token from [here](#) for a specific version.

### Invalid token

If the application is registered with an invalid token, another version of license token, or another platform's license token, the following error message will pop up when launching the application.

#### Error Message:

The included Bold license is invalid. Please refer to this help topic <https://documentation.boldreports.com/licensing/licensing-errors/#invalid-token> for more information.

#### Solution:

Generate a valid license token from [here](#) for a specific version.

### License Expired

The following error message will be shown if the license token has expired.

#### Error Message:

Your Bold license has expired. Please refer to this help topic <https://documentation.boldreports.com/licensing/licensing-errors/#license-expired> for more information.

**Solution:**

Purchase from [here](#) to get Bold license.

### Platform Mismatch

If the application is registered with another platform's license token, the following error message will pop up when launching the application.

**Error Message:**

The included Bold license is invalid (Platform mismatch). Please refer to this help topic <https://documentation.boldreports.com/licensing/licensing-errors/#platform-mismatch> for more information.

**Solution:**

Generate a valid license token from [here](#) platform.

### Internet Connection Error

We need internet connection to validate the registered license. If internet is not connected in the machine, the following error message will pop up when launching the application.

**Error Message:**

We couldn't connect to internet to validate your license. Please check your internet connection.

**Solution:**

Connect the internet and try running the application again.

### Network Error

If the server is not reachable to validate the license token, the following error message will pop up when launching the application.

**Error Message:**

Unable to validate your license. Please try again after some time or check your firewall/proxy settings. Refer to the help topic <https://documentation.boldreports.com/licensing/licensing-errors/#network-error>.

**Solution:**

Please ensure that you are connected to the internet. If that does not resolve the problem, check your proxy settings.

### Could not load Bold.Licensing.dll assembly version

Make sure that all the referenced Bold assemblies are of the same version. Try cleaning and rebuilding the application to resolve assembly conflict issues.

## FAQ section for Bold Licensing

We are going to discuss about most frequently asked questions of Bold Licensing in this section. We discussed the below topics in this section,

[How to upgrade from Trial version after purchasing a license?](#)

[Where can I get a license token?](#)

[Bold Licensing before v2.x](#)

[Is Bold Reports Embedded Reporting Tools or Viewer SDK requires additional licensing when running application with Azure App service?](#)

## How to upgrade from Trial version after purchasing a license

Replace the currently used trial license token with a paid license token that can be generated from the [License & Downloads](#) section of our Bold Reports website. Refer to [this](#) topic for more information regarding registering the license in the application.

### Where can I get a license token

License tokens can be generated from the [License & Downloads](#) section of the Bold Reports website. Refer to [this](#) topic for more information to generate the license token in the application.

## Does Bold Reports Embedded Reporting Tools or Viewer SDK require additional licensing when deploying an application to Azure App service

No, if you already have Bold Reports for your team, you do not need to buy an additional license for deploying your applications to Azure App service. You should register the license token on initial startup of your application as explained in the following documentation.

[License Token](#)

### See also

[What are the differences in Bold Reports licensing models](#)

[Can Syncfusion licenses be used with Bold Reports](#)

[Can the Syncfusion community license be used with Bold Reports](#)

[Can the Report Viewer component be accessed from Bold Reports using Syncfusion community license](#)

[Can the Report Designer component be accessed from Bold Reports using Syncfusion community license](#)

[Can the Report Designer component be used from Bold Reports using Syncfusion license](#)

[Is Report Designer included with Report Viewer SDK to create Reports](#)

## Can Syncfusion licenses be used with Bold Reports

Yes, you can use Syncfusion licenses with Bold Reports, but with access only to Report Viewer SDK. You should login with Syncfusion account to access Report Viewer SDK. To know more about licensing, please refer to [What are the differences in Bold Reports licensing models](#).

See also

[Can the Report Designer component be used from Bold Reports using Syncfusion license](#)

## Can the Syncfusion community license be used with Bold Reports

Yes, you can use the Syncfusion community license with Bold Reports, with access only to Report Viewer SDK. You should login with Syncfusion account to access Report Viewer SDK. To know more about licensing, check [What are the differences in Bold Reports licensing models](#).

See also

[Can the Report Viewer component be accessed from Bold Reports using Syncfusion community license](#)

[Can the Report Designer component be accessed from Bold Reports using Syncfusion community license](#)

## Can the Report Viewer component be accessed from Bold Reports using Syncfusion community license

Yes, you can access the Report Viewer component from Bold Reports using Syncfusion Community license. You should login with Syncfusion account to access the component. For more details about the licensing, refer to [What are the differences in Bold Reports licensing models](#).

See also

[Can the Syncfusion community license be used with Bold Reports](#)

[Can the Report Designer component be accessed from Bold Reports using Syncfusion community license](#)

## Can the Report Designer component be accessed from Bold Reports using Syncfusion community license

No, you cannot access the Report Designer component from Bold Reports by using Syncfusion community license. You should purchase the Embedding Reporting license with Bold Reports for using the Report Designer component. For more details about the licensing, refer to [What are the differences in Bold Reports licensing models](#).

See also

[Can the Syncfusion community license be used with Bold Reports](#)

[Can the Report Viewer component be accessed from Bold Reports using Syncfusion community license](#)

## Can the Report Designer component be used from Bold Reports using Syncfusion license

No, you cannot use the Report Designer component from Bold Reports by using Syncfusion license. You should purchase the Embedding Reporting license with Bold Reports for using Report Designer component. To know more about licensing, please refer to [What are the differences in Bold Reports licensing models](#).

See also

[Can Syncfusion licenses be used with Bold Reports](#)

## Is Report Designer included with Report Viewer SDK to create Reports

Yes, Standalone Report Designer is included with Report Viewer SDK license for creating reports. You can create reports with Standalone Report Designer application and save it in your application or share the network location for using them with the Report Viewer and Report Writer.

If you want to add Report Designer component to your application (for create and edit functionalities), you should purchase the Embedding Reporting license with Bold Reports. To know more about licensing, please refer to [What are the differences in Bold Reports licensing models](#).

## What are the differences in Bold Reports licensing models

Bold Reports has different restrictions when using reporting tools and solutions with licensing models. Refer to the following table more details.

| License Models                                                                   | Cloud Reporting | Enterprise Reporting |
|----------------------------------------------------------------------------------|-----------------|----------------------|
| Embedded Reporting   Report Viewer SDK                                           |                 |                      |
| Bold Reports Cloud Report Server                                                 | Yes             | No                   |
| No                                                                               |                 |                      |
| Bold Reports Report Server                                                       | No              | Yes                  |
| No                                                                               |                 |                      |
| Report Designer for application                                                  | No              | Yes                  |
| No                                                                               |                 |                      |
| Report Viewer for application                                                    | No              | Yes                  |
| Yes                                                                              |                 |                      |
| Report Writer for application (Export Reports to different formats without view) | No              | No                   |
| Yes                                                                              | Yes             |                      |
| Standalone Report Designer for creating Reports                                  | Yes             | Yes                  |
| Yes                                                                              | Yes             |                      |

## See also

[Can Syncfusion licenses be used with Bold Reports](#)

[Can the Syncfusion community license be used with Bold Reports](#)

[Can the Report Viewer component be accessed from Bold Reports using Syncfusion community license](#)

[Can the Report Designer component be accessed from Bold Reports using Syncfusion community license](#)

[Can the Report Designer component be used from Bold Reports using Syncfusion license](#)

[Is Report Designer included with Report Viewer SDK to create Reports](#)

[Does Bold Reports Embedded Reporting Tools or Viewer SDK require additional licensing when deploying an application to Azure App service](#)

## Embedded reporting tools licensing version 1.2.x

If you are using [Bold Reports](#) lesser than the version 2.x then you need to register the license key with your application. In this section we discussed about the below topics,

[How to register a license key](#)

[Licensing errors while using v1.2.x of Bold Reports.](#)

This section specifically for Bold Reports version less than 2.x. If you are using higher version of [Bold Reports](#), we recommend you to refer the section [licensing tokens](#) to register your application with Bold Licensing.

## Embedded reporting tools licensing overview

Starting with the version 1.2.x, Syncfusion introduced a new licensing system for the Bold Reports. You have to include the license key in your projects in order to use the Bold Reports assemblies. Note that the license key is different from the setup unlock key and that needs to be generated separately from the Bold Reports website. The below licensing error will be displayed if the license key is missing in your projects.

This application was built using a trial version of Bold Reports. Please include a valid license to permanently remove this license validation message. You can also obtain a free 30 day evaluation license to temporarily remove this message during the evaluation period.

[How to generate Bold Reports license key](#)

License keys can be generated from the [Downloads](#) section of the [Bold Reports](#) site.

Bold Reports license keys are version specific. So, you should use the corresponding version license key in the projects.

## How to register the Bold Reports license key

The generated license key is just a string that needs to be registered before any Bold Reports control is initiated. The following code is used to register the license.

'csharp

```
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE KEY");
'
```

## ASP.NET

Register the license key in Application\_Start method of `Global.asax.cs/Global.asax`

'csharp

```
void Application_Start(object sender, EventArgs e)
{
 //Register Bold license

 Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE KEY");
 // Code that runs on application startup

 RouteConfig.RegisterRoutes(RouteTable.Routes);
 BundleConfig.RegisterBundles(BundleTable.Bundles);
}
```

## ASP.NET MVC

Register the license key in Application\_Start method of `Global.asax.cs`

'csharp

```
void Application_Start(object sender, EventArgs e)
{
 //Register Bold license

 Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE KEY");
 // Code that runs on application startup

 RouteConfig.RegisterRoutes(RouteTable.Routes);
 BundleConfig.RegisterBundles(BundleTable.Bundles);
}
```

## ASP.NET Core

Register the license key in `Configure` method of `Startup.cs`

'csharp

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env, ILoggerFactory loggerFactory)
```

```
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE KEY");
loggerFactory.AddConsole(Configuration.GetSection("Logging"));
loggerFactory.AddDebug();
}
`
```

### WPF

Register the license key in App constructor of `App.xaml.cs` in C#. If App constructor not available in `App.xaml.cs`, create the `App()` constructor in `App.xaml.cs` and register the license key inside the constructor.

```
`csharp
public partial class App : Application
{
public App()
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE KEY");
}
}
`
```

### UWP

Register the license key in `App.xaml.cs` constructor before `InitializeComponent()` in C#. If App constructor not available in `App.xaml.cs`, create the `App()` constructor in `App.xaml.cs` and register the license key inside the constructor.

```
`csharp
public App()
{
//Register Bold license
Bold.Licensing.BoldLicenseProvider.RegisterLicense("YOUR LICENSE KEY");
this.InitializeComponent();
this.Suspending += OnSuspending;
}
`
```

## Embedded reporting tools license key errors

Licensing error popup is displayed with various messages under different circumstances. Here are some ways to resolve different issues.

### License key not registered

The following error message will be shown if Bold license key was not registered in your application.

#### Error message:

This application was built using a trial version of Bold Reports. Please include a valid license to permanently remove this license validation message. You can also obtain a free 30 day evaluation license to temporarily remove this message during the evaluation period. Please refer to this <https://documentation.boldreports.com/licensing/faq/v1.x/> help topic for more information.

#### Solution:

Generate a valid license key from [here](#) for a specific version.

### Invalid key

If the application is registered with an invalid key, another version of license key, or another platform's license key, the following error message will pop up when launching the application.

#### Error Message:

The included Bold license is invalid. Please refer to this help topic

<https://documentation.boldreports.com/licensing/faq/v1.x/errors/#invalid-key> for more information.

#### Solution:

Generate a valid license key from [here](#) for a specific version.

### Trial Expired

The following error message will be shown if the trial key has expired after 30 days.

#### Error Message:

Your Bold trial license has expired. Please refer to this help topic

<https://documentation.boldreports.com/licensing/faq/v1.x/errors/#trial-expired> for more information.

#### Solution:

Purchase from [here](#) to get a valid Bold license.

### License Expired

The following error message will be shown if the license key has expired.

#### Error Message:

Your Bold license has expired. Please refer to this help topic

<https://documentation.boldreports.com/licensing/faq/v1.x/errors/#license-expired> for more information.

#### Solution:

Purchase from [here](#) to get Bold license.

## Platform Mismatch

If the application is registered with another platform's license key, the following error message will pop up when launching the application.

### Error Message:

The included Bold license is invalid (Platform mismatch). Please refer to this help topic <https://documentation.boldreports.com/licensing/faq/v1.x/errors/#platform-mismatch> for more information.

### Solution:

Generate a valid license key from [here](#) platform.

## Version Mismatch

If the application is registered with another version's license key, the following error message will pop up when launching the application.

### Error Message:

The included Bold license ({Registered Version}) is invalid for version {Required version}. Please refer to this help topic <https://documentation.boldreports.com/licensing/faq/v1.x/errors/#version-mismatch> for more information.

### Solution:

Generate a valid license key from [here](#) platform.

## Could not load Bold.Licensing.dll assembly version

Make sure that all the referenced Bold assemblies are of the same version. Try cleaning and rebuilding the application to resolve assembly conflict issues.