Sage 300 Web Screens SDK

Declarative Setup Reports Interface

August 2022

The MIT License (MIT)

Copyright © 2022 The Sage Group plc or its licensors. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Contents

[1. Introduction 4](#_Toc105498509)

[2. How to add a report 5](#_Toc105498510)

[2.1 File Path 5](#_Toc105498511)

[2.2 JSON Configuration 5](#_Toc105498512)

[2.3 Menu Screen URL 6](#_Toc105498513)

[3. Examples 7](#_Toc105498514)

[3.1 Simple report with no parameters 7](#_Toc105498515)

[3.2 Report with multilanguage support 8](#_Toc105498516)

[3.3 Report with custom layout 10](#_Toc105498517)

[3.4 Report with option type controls 12](#_Toc105498518)

[3.5 Report with Section control 14](#_Toc105498519)

[3.6 AP Account Set Report 16](#_Toc105498520)

[4. JSON Configuration 18](#_Toc105498521)

[4.1 Root Properties 18](#_Toc105498522)

[4.2 Parameters 19](#_Toc105498523)

[4.3 ControlType 20](#_Toc105498524)

[4.4 Range 20](#_Toc105498525)

[4.5 DataSource 21](#_Toc105498526)

[4.6 HtmlAttrs 21](#_Toc105498527)

[4.7 Textbox 21](#_Toc105498528)

[4.8 Finder 21](#_Toc105498529)

[4.9 Numeric 22](#_Toc105498530)

[4.10 Datepicker 22](#_Toc105498531)

[4.11 Checkbox 22](#_Toc105498532)

[4.12 RadioButtonGroup 22](#_Toc105498533)

[4.13 Dropdown 23](#_Toc105498534)

[4.14 Section 23](#_Toc105498535)

[4.15 RenderDirection 23](#_Toc105498536)

1. Introduction

This document is intended to provide information on the declarative setup report interface, introduced in Sage 300 2023, which allows developers to create a report web screen via a JSON configuration as opposed to regular web screen development consisting of managed code, Razor views and JavaScript files.

1. How to add a report

The assumption is that the Crystal Report and supporting Accpac View already exist.

* 1. File Path

The path to the JSON configuration is configurable in the web.config file:

<add key="DeclarativeReportLocation" value="~/App\_Data/Reports" />

The default is **"~/App\_Data/Reports**" and both absolute and relative paths are supported.

Ensure the above setting in the web.config file exists, and the file path specified exists as this is where the JSON configuration file will reside.

* 1. JSON Configuration

A JSON configuration contains one report and can be found in the above noted file path in a ***module Id*** subfolder:

Graphical user interface, text, application

Description automatically generated

The folder hierarchy is for segregation of reports and to accommodate 3rd party reports

Refer to the examples in this document for creating the JSON configuration.

* 1. Menu Screen URL

The URL for a declarative report is specified in the module’s XML menu file:

<ScreenURL>Core/DeclarativeReport/Load?module=AP&amp;report=APAccountSet</ScreenURL>

The ***module*** parameter is optional and refers to the folder under the above noted file path. Multiple subfolders may be defined in the module parameter (i.e., ValuedPartner1/AP).

The ***report*** parameter is required and refers to the name of the JSON configuration file without extensions.

An example of the AP Account Sets declarative setup report’s menu item:

Graphical user interface, text, application

Description automatically generated

The modules XML menu file will require a new entry specifying the report to be menu linked

1. Examples

Declarative Setup Report configurations for ~97 declarative setup reports for Sage 300 can be found in your Sage 300 web screen installation folder, in the path defined by the ***DeclarativeReportLocation*** setting in web.config file. These are also great examples to be leveraged when creating declarative setup reports.

The examples listed in this document cover most configurations of Sage 300 declarative setup reports to-date.

* 1. Simple report with no parameters

This example report has no parameters. It is a very simple report that has no user input required for the report.

* + 1. Configuration

|  |
| --- |
| {  "ReportNames": [ "APACCT01" ], // Comma separated list of multiple reports  "ReportName": "APACCT01", // Default report  "EntityName": "AP0006", // ACCPAC entity for the report  "ScreenNameText": "Simple report", // Report title (not localized)  "Module": "AP", // Report’s module  "SecurityResource": "APCOMUI", // User permission required for the report  } |

* + 1. Screen Example



* 1. Report with multilanguage support

This example report has multilanguage support. It is a very simple report that has a couple of user inputs and supports localization.

* + 1. Configuration

|  |
| --- |
| {  "ReportNames": [ "APACCT01"],  "ReportName": "APACCT01",  "EntityName": "AP0006",  "ScreenNameResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AP\_AccountSetsTitle", // Report title (localized)  "Module": "AP",  "SecurityResource": "APCOMUI",  "ResourceAssembly": "Sage.CA.SBS.ERP.Sage300.Common.Resources.dll", // Resource assembly for resource strings for report  "ResourceAssemblyPath": "~\\bin\\", // Location of the resource assembly  "Parameters": [  {  "Name": "FROMCODE",  "Length": "6",  "ControlType": "Finder",  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetFrom", // Resource string for the textbox label  "ID": "txtFromAcct",  "ButtonID": "btnFromAcct",  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  },  {  "Name": "TOCODE",  "Length": "6",  "ControlType": "Finder",  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetTo", // Resource string for the textbox label  "ID": "txtToAcct",  "ButtonID": "btnToAcct",  "DefaultValue": "ZZZZZZ",  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  }  ]  } |

* + 1. Screen Example

Graphical user interface, text, application

Description automatically generated

* 1. Report with custom layout

This example report has custom layout support. It is a simple report that has a couple of user inputs, supports localization, and a custom layout.

* + 1. Configuration

|  |
| --- |
| {  "ReportNames": [ "APACCT01" ],  "ReportName": "APACCT01",  "EntityName": "AP0006",  "ScreenNameResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AP\_AccountSetsTitle",  "Module": "AP",  "SecurityResource": "APCOMUI",  "ResourceAssembly": "Sage.CA.SBS.ERP.Sage300.Common.Resources.dll",  "ResourceAssemblyPath": "~\\bin\\",  "HtmlTemplate": "<div>{resx:Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSet}<div>{txtFromAcct}</div>  <div>{txtToAcct}</div></div>", // Custom layout with a resource string and 2 parameters specified by ID  "Parameters": [  {  "Name": "FROMCODE",  "Length": "6",  "ControlType": "Finder",  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetFrom",  "ID": "txtFromAcct", // ID of the textbox  "ButtonID": "btnFromAcct",  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  },  {  "Name": "TOCODE",  "Length": "6",  "ControlType": "Finder",  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetTo",  "ID": "txtToAcct", // ID of the textbox  "ButtonID": "btnToAcct",  "DefaultValue": "ZZZZZZ",  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  }  ]  } |

* + 1. Screen Example

A picture containing application

Description automatically generated

* 1. Report with option type controls

This example report has controls presenting various options. It is a simple report that has user inputs with options.

* + 1. Configuration

|  |
| --- |
| {  "ReportNames": [ "APACCT01" ],  "ReportName": "APACCT01",  "EntityName": "AP0006",  "ScreenNameResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AP\_AccountSetsTitle",  "Module": "AP",  "SecurityResource": "APCOMUI",  "ResourceAssembly": "Sage.CA.SBS.ERP.Sage300.Common.Resources.dll",  "ResourceAssemblyPath": "~\\bin\\",  "Parameters": [  {  "Name": "SortBy",  "ControlType": "Dropdown", // Dropdown control  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.CommonResx.SortBy",  "ID": "ddlSortBy",  "DefaultValue": "ITEMNO", // Initial selection  "Size": "medium",  "DataSource": [ // Data source for dropdown  {  "Value": "ITEMNO", // Option value  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.ItemNumber" // Option text  },  {  "Value": "CATEGORY",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.Category"  },  {  "Value": "SEGMENT",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.ItemSegment"  }  ]  },  {  "Name": "SWINCLCOMPBOMS",  "ControlType": "Checkbox", // Checkbox control  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.IncludeComponentBOMs",  "ID": "chkIncludeCompBoms",  "DefaultValue": "0", // Initial selection  "DataSource": [ // Data source for checkbox  {  "Value": "0" // First option is the false value  },  {  "Value": "1" // Second option is the true value  }  ]  }  ]  } |

* + 1. Screen Example

Graphical user interface, application

Description automatically generated with medium confidence

* 1. Report with Section control

This example report has controls contained in a section. It is a simple report that has user inputs for a range.

* + 1. Configuration

|  |
| --- |
| {  "ReportNames": [ "APACCT01" ],  "ReportName": "APACCT01",  "EntityName": "AP0006",  "ScreenNameResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AP\_AccountSetsTitle",  "Module": "AP",  "SecurityResource": "APCOMUI",  "ResourceAssembly": "Sage.CA.SBS.ERP.Sage300.Common.Resources.dll",  "ResourceAssemblyPath": "~\\bin\\",  "Parameters": [  {  "Name": "Panel", // Name of section  "Active": false, // Do not pass this parameter to the Crystal Report  "ID": "panel",  "ControlType": "Section", // Section control used to contain other controls  "DisplayBorders": false, // Do not show top and bottom borders of the panel  "RenderingDirection": "Horizontal" // Controls inside are displayed horizontally  },  {  "Name": "FROMCODE",  "Length": "6",  "ControlType": "Finder",  "ParentContainer": "Panel", // Name of section  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetFrom",  "ID": "txtFromAcct",  "ButtonID": "btnFromAcct",  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  },  {  "Name": "TOCODE",  "Length": "6",  "ControlType": "Finder",  "ParentContainer": "Panel", // Name of section  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetTo",  "ID": "txtToAcct",  "ButtonID": "btnToAcct",  "DefaultValue": "ZZZZZZ",  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  }  ]  } |

* + 1. Screen Example

Graphical user interface, application

Description automatically generated

* 1. AP Account Set Report

This example report is the AP Account Set report for Sage 300. It is a simple report that has numerous features such as multiple Crystal report formats, localization, custom JavaScript, range validation, multi-currency checks, etc.

* + 1. Configuration

|  |
| --- |
| {  "ReportNames": [ "APACCT01", "APACCT1Z" ], // Allowed reports via this web screen  "ReportName": "APACCT01", // Default report  "EntityName": "AP0006", // ACCPAC entity  "ScreenNameResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AP\_AccountSetsTitle", // Resource string for the report title  "Module": "AP", // Report’s Module  "SecurityResource": "APCOMUI", // User permission required  "ResourceAssembly": "Sage.CA.SBS.ERP.Sage300.Common.Resources.dll", // Resource assembly to load for resource strings  "ResourceAssemblyPath": "~\\bin\\", // Location of the resource assembly  "CustomScripts": [ // Javascript files to include  "~/Areas/Core/Scripts/DeclarativeReport/Shared/Sage.CA.SBS.ERP.Sage300.Common.DeclarativeReport.SetMultiCurrencyYN.js", // This Javascript file contains a function to set the MULTCURN? parameter to “Y” if AP multicurrency option is true  "~/Areas/Core/Scripts/DeclarativeReport/Shared/Sage.CA.SBS.ERP.Sage300.Common.DeclarativeReport.SetFunctionalCurrency.js", // This Javascript file contains a function to set the FUNCCURN parameter to the functional currency code in company options  "~/Areas/AP/Scripts/Reports/AccountSet/Sage.CA.SBS.ERP.Sage300.AP.AccountSetBehaviour.js" // This javascript file contains a function to set the ReportName if GL module is active  ],  "Ranges": [ // Paired parameters in this list will be rendered side by side. When print is clicked, an error message will appear if the value of the From parameter is greater than the value of the To parameter  {  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSet", // Used in the error message  "From": "FROMCODE", // Parameter name of the From range  "To": "TOCODE" // Parameter name of the To range  }  ],  "Parameters": [ // Controls that will be rendered on the page  {  "Name": "FROMCODE", // Parameter name in the Crystal report  "Length": "6", // Maximum length of the textbox value  "ControlType": "Finder", // A finder will be rendered  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet", // The Javascript object containing the finder definition  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetFrom", // Resource string for the textbox label  "ID": "txtFromAcct", // ID of the textbox  "ButtonID": "btnFromAcct", // ID of the finder button  "HtmlAttrs": [ // Html attributes of the textbox  {  "Key": "formattextbox",  "Value": "alphaNumeric" // Textbox only accepts alphanumeric characters  }  ]  },  {  "Name": "TOCODE",  "Length": "6",  "ControlType": "Finder",  "FinderDefinition": "sg.viewFinderProperties.AP.AccountSet",  "LabelResource": "Sage.CA.SBS.ERP.Sage300.Common.Resources.DeclarativeReportResx.AccountSetTo",  "ID": "txtToAcct",  "ButtonID": "btnToAcct",  "DefaultValue": "ZZZZZZ", // Initial value of the textbox when the page is loaded  "HtmlAttrs": [  {  "Key": "formattextbox",  "Value": "alphaNumeric"  }  ]  },  {  "Name": "MULTCURN?",  "ControlType": "Hidden", // A hidden input will be rendered  "ID": "hdnMulticurrency",  "DefaultValue": "N",  "IsMultiCurrency": true, // This parameter should be validated on the server side for multicurrency  "ValidationErrorValue": "N" // If multicurrency validation fails, this value is passed instead to print  },  {  "Name": "FUNCCURN",  "ControlType": "Hidden",  "ID": "hdnFunctionalCurrency"  }  ]  } |

* + 1. Screen Example

Graphical user interface, application

Description automatically generated

1. JSON Configuration

The following section will describe the format of the JSON configuration file.

* 1. Root Properties

The root section of a report JSON file.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| ReportName | string | Checkmark with solid fill | Crystal report file name to print, excluding extension |
| ReportNames | string[] | Checkmark with solid fill | List of valid Crystal reports the web screen can print. This must contain the **ReportName** |
| EntityName | string | Checkmark with solid fill | Business entity name (Accpac View) |
| Module | string | Checkmark with solid fill | Module ID |
| ScreenNameText | string |  | Web screen title specified if no localization is required |
| ScreenNameFunction | string |  | Javascript function that returns the web screen title. If set, overrides **ScreenNameText** and **ScreenNameResource** |
| ScreenNameResource | string |  | Resource string that returns the web screen title. If set, overrides **ScreenNameText** |
| ResourceAssembly | string |  | Assembly to load resource strings for supporting localization |
| ResourceAssemblyPath | string |  | Resource assembly path |
| HtmlTemplate | string |  | Html layout for parameters. Used if a different layout other than the generated one is desired. Controls can be injected into the template marked by {Parameter.**ID**}. Resource strings can be injected into the template marked by {resx:resourceBaseName.resourceName} |
| SecurityResource | string |  | Security string to check if user has access to the report. This should match the value of the **SecurityResourceKey** in the web screen menu’s XML file |
| Parameters | List<Parameter> |  | List of parameters for the report |
| Ranges | List<Range> |  | List of ranges in Parameters. Parameter pairs defined will be rendered horizontally, and the values will be checked when print is clicked. If the **From** parameter value is greater than the **To** parameter value, an error will display |
| CustomScripts | List<string> |  | List of Javascript files to be included in the page. The path and file must exist and be available to the web application. Only .js files are accepted. This can be used to add additional logic to the web screen |
| RazorViews | List<string> |  | List of Razor files to be included in the page. The path and file must exist and be available to the web application. Only .cshtml files are accepted. |

* 1. Parameters

An array of parameters containing generated controls for the page which are usually mapped to a parameter in the Crystal Report.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| ID | string | Checkmark with solid fill | Html id attribute of the control |
| Name | string | Checkmark with solid fill | Crystal Report parameter name |
| ControlType | ControlType | Checkmark with solid fill | Type of UI control (Textbox, Finder, Dropdown, Checkbox, Hidden, Numeric, Datepicker, RadioButtonGroup, Currency, Section) |
| Value | string |  | Value sent to the Crystal Report |
| Length | int |  | Maximum length for **Textbox**, **Finder**, **Numeric**, and **Currency** ControlType parameters |
| Hidden | bool |  | True to hide the parameter. Default **false** |
| ParentContainer | string |  | **Name** of a **Section** ControlType parameter that contains this parameter |
| HasCustomBinding | bool |  | True to add Knockout attributes. Default **false** |
| DataAttrs | List<KeyValue> |  | Data attributes for the control, used by Knockout binding |
| HtmlAttrs | List<KeyValue> |  | Html attributes for the control |
| DefaultValue | string |  | Initial value when the page is loaded |
| Datasource | List<DataSource> |  | Static data source for **Dropdown**, **Checkbox**, and **RadioButtonGroup** ControlType parameters. Dynamic data sources can be initialized using Javascript defined from **CustomScripts** |
| IncludeLabel | bool |  | True to render a label with the control. Default **true** |
| LabelText | string |  | Label text |
| LabelFunction | string |  | Javascript function that returns the label text. If set, overrides **LabelText** and **LabelResource** |
| LabelResource | string |  | Resource string that returns the label text. If set, overrides **LabelText** |
| LabelDataAttrs | List<KeyValue> |  | Data attributes for the label, used by Knockout binding |
| LabelHtmlAttrs | List<KeyValue> |  | Html attributes for the label |
| Active | bool |  | True to send the parameter to the Crystal Report. Default **true**. Parameters used for formatting and helpers should be set to false |
| SecurityResource | string |  | Security string to check if user has access to the parameter. If not, the parameter is not rendered |
| ValidationErrorValue | string |  | Used for server validation. If **IsOptionalFields** or **IsMultiCurrency** is true, or **LicensesToValidate** is not empty, these options and licenses are checked on print. If the result is false, **ValidationErrorValue** is sent to the Crystal Report instead of the **Value** passed by the client |
| IsOptionalFields | bool |  | True if the parameter requires optional fields license. If there is no license, the parameter will be hidden. Default **false**. On print, if there is no license, **ValidationErrorValue** is sent to the Crystal Report instead of the client’s supplied **Value** |
| IsMultiCurrency | bool |  | True if the parameter should validate on print the module’s multicurrency option. Default **false**. If multicurrency is false, **ValidationErrorValue** is sent to the Crystal Report instead of the client’s supplied **Value** |
| LicensesToValidate | string[] |  | List of licenses to validate on print. If any of the licenses are not active, **ValidationErrorValue** is sent to the Crystal Report instead of the client’s supplied **Value** |
| IncludeValidation | bool |  | True to include html attributes for client-side validation. Default **true** |
| ValidationText | string |  | The text message for client-side validation |

* 1. ControlType

Enum of supported control types for a parameter. Refer to Sage300SDK\_HtmlHelpers.docx for details

|  |  |  |
| --- | --- | --- |
| **Name** | **Enum** | **Description** |
| Textbox | 0 | A textbox |
| Finder | 1 | A finder |
| Dropdown | 2 | A dropdown |
| Checkbox | 3 | A checkbox |
| Hidden | 4 | A hidden input |
| Numeric | 5 | A numeric textbox |
| Datepicker | 6 | A datepicker |
| RadioButtonGroup | 7 | A group of radio buttons |
| Currency | 8 | A currency textbox |
| Section | 9 | A container for other Control Types |

* 1. Range

Parameter pairs defined will be rendered horizontally, and the values will be checked when print button is clicked. If the From parameter value is greater than the To parameter value, an error message will display, with the LabelText referring to the type of the value in the message.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| From | string | Checkmark with solid fill | From parameter **Name** |
| To | string | Checkmark with solid fill | To parameter **Name** |
| LabelText | string |  | Label text |
| LabelFunction | string |  | Javascript function that returns the label text. If set, overrides **LabelText** and **LabelResource** |
| LabelResource | string |  | Resource string that returns the label text. If set, overrides **LabelText** |

* 1. DataSource

Static data source for Dropdown, Checkbox, and RadioButtonGroup ControlType parameters. Dynamic data sources can be initialized using Javascript defined from CustomScripts. LabelText is the display text of the option.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| Value | string | Checkmark with solid fill | Data source value |
| LabelText | string |  | Label text |
| LabelFunction | string |  | Javascript function that returns the label text. If set, overrides **LabelText** and **LabelResource** |
| LabelResource | string |  | Resource string that returns the label text. If set, overrides **LabelText** |

* 1. HtmlAttrs

Dictionary for Html attributes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| Key | string | Checkmark with solid fill | Key |
| Value | object |  | Value |

* 1. Textbox

Inherits Parameter (see 4.2). These are additional properties for the control type.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| Size | string |  | The CSS class for the size of the textbox. Default **‘default’** (default, small, smaller, xsmall, large, larger, xlarge, medium, mediumlarge) |

* 1. Finder

Inherits Parameter (see 4.2). These are additional properties for the control type.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| ButtonID | string | Checkmark with solid fill | Html id attribute of the button |
| FinderDefinition | string | Checkmark with solid fill | Finder properties. Refer to Sage300SDK\_FinderArchitecture.docx for details |
| Size | string |  | The CSS class for the size of the textbox. Default **‘default’** (default, small, smaller, xsmall, large, larger, xlarge, medium, mediumlarge) |
| IsNumeric | bool |  | True if the values of the type of finder are numeric. Default **false** |
| GoButtonID | string |  | Html id attribute of the Go button |
| LabelWithHamburger | bool |  | True to include a hamburger next to the label. Default **false** |
| HamburgerUrl | string |  | Url of the hamburger control |
| HamburgerValue | string |  | Value of the hamburger control |
| ButtonDataAttrs | List<KeyValue> |  | Data attributes for the button, used by Knockout binding |
| IncludeNavigationButtons | bool |  | True to include navigation buttons (to navigate to First, Previous, Next, or Last record). Default **false** |
| IncludeAddButton | bool |  | True to include the add button (to create new record). Default **false** |
| NavGroupIndex | int |  | Navigation group index |
| IncludeGoButton | bool |  | True to include the Go button. Default **false** |
| IncludeFinderButton | bool |  | True to include the button to open the finder popup. Default **true** |

* 1. Numeric

Inherits Parameter (see 4.2). These are additional properties for the control type.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| DecimalPlaces | int |  | Number of decimal places |
| ShowSpinners | bool |  | True to display a spinner. Default **false** |
| StepIncrement | int |  | Step increment |
| MaxDigits | int |  | The maximum number of digits to display |
| MinValue | int |  | The minimum allowable value |
| MaxValue | int |  | The maximum allowable value |
| FormatString | string |  | The formatting string template |
| Size | string |  | The CSS class for the size of the textbox. Default **‘default’** (default, small, smaller, xsmall, large, larger, xlarge, medium, mediumlarge) |

* 1. Datepicker

Inherits Parameter (see 4.2). These are additional properties for the control type

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| Size | string |  | The CSS class for the size of the textbox. Default **‘small’** (default, small, smaller, xsmall, large, larger, xlarge, medium, mediumlarge) |

* 1. Checkbox

Inherits Parameter (see 4.2). These are additional properties for the control type

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| LabelFirst | bool |  | True to display the label on the left of the checkbox. False to display the label on the right of the checkbox. Default **false** |

* 1. RadioButtonGroup

Inherits Parameter (see 4.2). These are additional properties for the control type

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| LabelFirst | bool |  | True to display the label on the left of the radio button. False to display the label on the right of the radio button. Default **false** |

* 1. Dropdown

Inherits Parameter (see 4.2). These are additional properties for the control type

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| OptionLabel | string |  | The option label for the dropdown |
| Size | string |  | The CSS class for the size of the textbox. Default **‘default’** (default, small, smaller, xsmall, large, larger, xlarge, medium, mediumlarge) |

* 1. Section

Inherits Parameter(see 4.2). These are additional properties for the control type.

Nested Sections are not supported.

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Type** | **Required** | **Description** |
| RenderingDirection | RenderDirection |  | Direction to lay out the controls in the panel. Default **‘Vertical’** |
| DisplayBorders | bool |  | True to show a top and bottom border around the panel. Default **true** |

* 1. RenderDirection

Enum of supported layouts for a Section ControlType parameter.

|  |  |  |
| --- | --- | --- |
| **Name** | **Enum** | **Description** |
| Vertical | 0 | Controls are laid out vertically |
| Horizontal | 1 | Controls are laid out horizontally |

1. Notes
   1. Parameter Security Resource and Custom HTML

In order to properly hide controls with security resources, some additional custom html is required. The typical control looks something like:

<div class='ctrl-group'>{chkHideFromNonManagers}</div>

This control needs to be hidden if the user doesn’t have (for example) ICMGR access:

“ID”: “chkHideFromNonManagers”,

….

“SecurityResource”: “ICMGR”,

In order to do this and not interfere with any system ID’s, wrap this in a span, and an element ID as indicated, using single quotes:

<span id='SecurityResource\_chkShowSinNumbers'>

<div class='ctrl-group'>{chkShowSinNumbers}</div></span>