

OpenText™ Exstream™

Designing for LiveEditor

Design and Production Documentation

Release 16.6.0

OpenText™ Exstream
Designing for LiveEditor
Rev.: 2019-Apr-30

This documentation has been created for software version 16.6.0.

It is also valid for subsequent software versions as long as no new document version is shipped with the product or is published at <https://knowledge.opentext.com>.

Open Text Corporation
275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111
Toll Free Canada/USA: 1-800-499-6544 International: +800-4996-5440
Fax: +1-519-888-0677

Support: <https://support.opentext.com>
For more information, visit <https://www.opentext.com>

Copyright © 2019 Open Text. All rights reserved.

Trademarks owned by Open Text.

One or more patents may cover this product. For more information, please visit, <https://www.opentext.com/patents>

Disclaimer

No Warranties and Limitation of Liability

Every effort has been made to ensure the accuracy of the features and techniques presented in this publication.
However, Open Text Corporation and its affiliates accept no responsibility and offer no warranty whether expressed or implied, for the accuracy of this publication.

Contents

Chapter 1: Getting Started Designing Interactive Documents	14
1.1 About Interactive Documents	14
1.1.1 What Interactive Documents Can Do For Your Business	14
1.1.2 Examples of Live Documents You Can Produce	15
1.1.3 About LiveEditor and Empower Editor	15
1.2 How Interactive Documents Work with Exstream	16
1.2.1 The Design Environment	17
1.2.2 The Editing Environment	18
1.2.3 The Processing Environment	19
1.2.4 Live and Empower Modules Available from Exstream	20
1.3 Creating Your First Live Document Application	22
1.3.1 Overview of the Live Document Application Development Process	23
1.3.2 Application Planning Considerations for Creating a Live Document	24
1.4 A Technical Look at the DLF File Structure	27
1.4.1 The DLF Folder Structure	29
1.4.2 Viewing and Accessing the Structure of a Live Document	31
Chapter 2: Managing Live Document Versions in an Enterprise Environment	32
2.1 Overview of Live Document Versioning	32
2.2 Allowing End Users to Edit Live Documents in Older Versions of LiveEditor	35
2.3 Requiring End Users to Upgrade	36
Chapter 3: Creating a Basic Live Document	38
3.1 Allowing End Users to Change Text	39
3.1.1 Allowing End Users to Make Free-Form Changes to Text	39
3.1.2 Making Free-Form Changes to Text: The End User Experience	41
3.1.3 Allowing End Users to Change Text Formatting	42
3.1.4 Allowing End Users to Change Text Properties	43
3.1.5 Changing Text Properties: The End User Experience	44
3.2 Allowing End Users to Change the Value of a Variable	45
3.3 Allowing End Users to Make Changes to a Table	48
3.3.1 Table Design Considerations for Live Documents	48

3.3.2 Allowing End Users to Edit Simple Tables	49
3.3.3 Allowing End Users to Edit Automated Tables	51
3.4 Allowing End Users to Add Images to a Page	54
3.5 Allowing End Users to Format Chart Text	56
3.6 Allowing End Users to Change the Size, Position, or Rotation of Objects	57
3.7 Allowing End Users to Reorder Documents Using the Outline Viewer	58
3.8 Defining Sections and Paragraphs for Live	59
3.8.1 Setting Up Section and Paragraph Property Inheritance	59
3.9 Setting Up a Live Document to Dynamically Import Text and Images at Run Time	60
3.9.1 Setting Up a Page to Dynamically Import Content	63
3.9.2 Converting Inline Image Placeholder Variables to Empty Image Objects	64
3.10 Using Flow Frames in Live Documents	65
3.11 Using Tables of Contents and Indexes in Live Documents	66
3.12 Designing Live Documents for Multiple Languages	68
3.12.1 Adding Language Layers to a Live Document	69
3.12.2 Creating Live Documents for Complex Text Languages	69
3.12.3 Creating DLF Output for Live Documents That Contain Multiple Language Layers	70
3.12.4 Adding Content to Language Layers in a Live Document	71
3.12.5 Using Language Layers in a Live Document: The End User Experience	72
3.13 Defining Layouts for Electronic Devices That Can Be Delivered from Live Documents	72
3.13.1 Editing and Producing Output from a Live Document That Includes Container Designs: The End User Experience	74
Chapter 4: Controlling End User Editing Using Selection Controls	76
4.1 Using Check Boxes and Radio Buttons to Illustrate Choices	80
4.1.1 Setting Up a Check Box	80
4.1.2 Setting Up a Radio Button	85
4.2 Using Drop-Down Lists to Show Options	88
4.3 Using an Image Selector to Allow End Users to Choose From a Set of Images	91
4.3.1 Including Images as Part of the DLF File	92
4.3.2 Dynamically Importing Images From a Shared Location	95
4.3.3 Using an Image Selector: The End User Experience	99
4.4 Using a Content Selection Group to Dynamically Select Content	100
4.4.1 Creating a Variable to Control the Content Selection Group	101
4.4.2 Creating the Objects to be Used in the Content Selection Group	102

4.4.3 Defining the Content Selection Group Settings	103
4.4.4 Locating Content Selection Group Variables	105
4.4.5 Using a Content Selection Group: The End User Experience	105
4.5 Using Buttons to Launch Actions	106
4.5.1 Adding a Button to a Page	107
4.5.2 Defining the Variable Settings of the Button	107
4.5.3 Customizing the Appearance of the Button	108
4.5.4 Specifying the Button Actions	110
4.6 Using a Calendar to Allow End Users to Choose Dates	111
4.7 Using Form Fields to Control Data Entry	113
4.7.1 Setting Up Pre-Made Form Fields	114
4.7.2 Adding Form Fields to a Page	116
4.7.3 Data Entry Mask Symbols	118
4.8 Setting Up a Variable Palette to Allow End Users to Insert Variables	119
4.9 Controlling the Visibility of Activation Buttons in LiveEditor	120
4.10 Changing the Fonts Used in LiveEditor Controls	121
Chapter 5: Using Live Tools to Create a More Intuitive and Automated Editing Experience	123
5.1 Using Interview Functionality to Drive Data Changes	123
5.1.1 Using Interview Documents in a Live Document	124
5.1.2 Using Interview Pages in a Live Document	125
5.1.3 Designating a Variable to be an Interview Variable	126
5.2 Using Design Layers to Provide Instructions to End Users	127
5.2.1 Controlling the Visibility of Design Layers in a Live Document	127
5.2.2 Controlling Whether Design Layers Are Printed When Producing Output from a Live Document	128
5.2.3 Disabling All Design Layers	129
5.3 Using Actions to Automate Editing Tasks	129
5.3.1 Setting Up an Action	129
5.3.2 Specifying an Icon to Display in a Toolbar	131
5.3.3 Adding Actions to Appear on the Live Menu	132
5.4 Executing Functions During LiveEditor Events	132
5.5 Executing Functions During Editing	134
5.6 Using Show/Hide to Simplify Editing	135
5.6.1 Controlling Whether End Users Can Show and Hide a Specific Object	136
5.6.2 Controlling How Hidden Objects Appear on the Outline Viewer	136

5.7 Using External Hyperlinks to Link to Locations Outside of a Live Document	137
5.7.1 Where to Find Information About Adding External Hyperlinks to Objects in Your Design	138
5.7.2 Controlling an End Users Interaction with External Hyperlinks in Live Documents	139
5.7.3 Adding Live Action Links to Text or Objects in a Live Document Using Designer	140
5.7.4 Adding Live Action Links to Paragraph Objects in a Live Document Using Design Manager	147
5.8 Using Internal Hyperlinks to Link to Locations Within a Live Document	149
5.8.1 Where to Find Information About Adding Internal Hyperlinks to Objects in Your Design	149
5.8.2 Best Practices For Including Internal Hyperlinks in Live Documents	150
5.9 Making Data Available in Live Documents	150
5.9.1 Embedding Data Files	151
5.9.2 Adding Authorized Data Files to a Live Application to Allow End Users to Access Data	153
5.9.3 Allowing End Users to Upload Distribution Lists	155
Chapter 6: Allowing End Users to Import/Export Components of a Live Document	156
6.1 Allowing End Users to Import Unformatted Text	157
6.2 Allowing End Users to Import and Export Formatted Text	158
6.2.1 Technical Requirements for Setting Up the Ability to Import/Export Formatted Content	159
6.2.2 Designing an Application to Allow Formatted Text to be Imported/Exported	162
6.3 Importing Pages from External Files into a Live Document	166
6.3.1 About Using Placeholder Documents to Import Pages into a Live Document	167
6.3.2 Automatically Importing External Pages into a Live Document	169
6.3.3 Allowing End Users to Import External Pages into a Live Document	173
6.3.4 Setting Up a Placeholder Document to Allow Imported DOCX or DXF Files to Flow	175
Chapter 7: Allowing End Users to Add and Modify Copies of Documents	178
7.1 Creating Recipient Data Pages to Allow End Users to Modify Recipient Information	180
7.2 Configuring Recipient Profiles for Live Documents	182
7.3 Including the Capability to Add Recipient Copies Based on an Action	183

7.4 Including Recipient Copies in Multiple Languages	184
7.5 Using Recipient Data Records to Populate Recipient Data From Outside of LiveEditor	184
7.6 Processing Considerations for Live Documents with Recipient Copies	185
7.7 Adding Recipient Copies: The End User Experience	185
Chapter 8: Using a Web Service to Exchange Data Between Live Documents and Other Enterprise Systems	188
8.1 Designing the Live Document to Support Data Submission	189
8.2 Mapping the Request Data	189
8.2.1 Submitting Content Using Attachments	190
8.3 Mapping the Response Data	191
8.4 Setting Up Success and Error Messages to Appear in LiveEditor After Web Service Calls	191
8.5 Using Single Sign-On Authentication for Live Documents Running on a Web Service	193
8.5.1 Setting Up an XML File to Identify Which Groups Can Access the Live Document	194
8.5.2 Adjusting the Authorization Settings on the Live Setting Object	195
Chapter 9: Collecting Data For a Live Document Using Web Forms	197
9.1 Creating an Interview Page to Collect Data	198
9.2 Best Practices for Designing a Document to Collect Data Using Web Forms	199
Chapter 10: Using Section Data with Live Documents	200
10.1 An Introduction to Using Section Data in a Live Application	200
10.2 How Section Data Affects Editing	201
10.3 Live Feature Support with Section Data	203
10.3.1 Disabling Document Combination for Section-Driven Live Documents	203
Chapter 11: Using XML Node Data with Live Documents	205
11.1 How XML Node-Driven Content Affects Editing in Live Documents	205
11.2 Live Feature Support with XML Nodes	206
11.2.1 Disabling Document Combination for XML Node-Driven Live Documents	207
Chapter 12: Managing Messaging and Marketing Content in Live Documents	208

12.1 Using Campaigns in Live Output	209
12.2 Setting a Campaign Priority	209
12.3 Controlling How Graphic Messages Are Placed in a Message Frame	210
12.4 Controlling How Messages Are Placed During Fulfillment	211
12.5 Adding Messaging and Marketing Content in LiveEditor: The End User Experience	211
Chapter 13: Using Barcodes in Live Documents	213
13.1 Types of Barcodes Supported in Live Documents	213
13.2 Special Considerations for Generating Barcodes for Live Documents	214
13.3 Special Considerations for Assigning Variables for Barcode Data in Live	215
Chapter 14: Using Tools for Providing Guidance to End Users	217
14.1 Using Themes to Visually Guide End Users	218
14.1.1 Creating and Setting Up Themes	219
14.1.2 Specifying How Themes are Used in LiveEditor	221
14.1.3 Controlling How Themes are Applied to Selection Controls	222
14.1.4 Previewing and Testing Themes	223
14.2 Providing Pop-Up Information to Describe Specific Objects	224
Chapter 15: Customizing the End User Interface and Navigation Options	226
15.1 Customizing the End User Interface	226
15.1.1 Using Views to Customize the End-User Interface	226
15.1.2 Setting Up a Custom Toolbar to Provide End Users with Quick Access to Common Tasks	233
15.2 Customizing Navigation	235
15.2.1 How LiveEditor Handles Basic Navigation between Editable Areas	235
15.2.2 Enabling Navigation to Editable Areas	236
15.2.3 Customizing the Order of Navigation Between Editable Areas	237
15.2.4 Creating Additional Navigation Controls	241
15.2.5 Allowing End Users to Navigate From the Table of Contents	241
15.2.6 Adding Automatic Navigation to Specific Interactive Areas	242
15.2.7 Customizing the Outline Viewer	243
Chapter 16: Controlling the Formatting Options Available to End Users	247
16.1 Controlling Font Behavior in Live Applications	248
16.1.1 Controlling the Fonts that End Users Can Use	249

16.1.2 Controlling How Fonts are Handled When Packaging a DLF File	251
16.2 Restricting the Colors Available to End Users	257
16.3 Using Style Sheets to Provide Pre-Set Formatting Options	258
Chapter 17: Controlling How End Users Save and Print Live Documents	261
17.1 Controlling End User Options for Saving a Live Document as a PDF	262
17.2 Defining the Properties of PDFs Created from Live Documents	262
17.3 Controlling How End Users Print a Live Document Locally	264
Chapter 18: Setting Up Tools to Review and Validate End Users' Changes	265
18.1 Setting Up Text Proofing Tools	265
18.1.1 Specifying Behavior for Spelling, Grammar, and Excluded Word Check ..	266
18.1.2 Supplying Custom Dictionaries for End Users	267
18.2 Setting Up Live Document Review Tools	267
18.2.1 Using Revision Tracking to Monitor Changes Made to a Live Document ..	269
18.2.2 Setting Up Revision Commenting to Capture Explanations for Changes ..	271
18.2.3 Tracking Revisions and Revision Comments in LiveEditor: The End User Experience	274
18.3 Validating Data Entered by End Users	284
18.3.1 Associating a Validation Variable with an Object	284
18.3.2 Specifying the Way End Users Are Notified of Invalid Data	285
Chapter 19: Controlling Access to Live Documents and Areas within Live Documents	286
19.1 Controlling Access to Live Documents	286
19.1.1 Defining the Authentication Settings	286
19.1.2 Defining Design Group Access	289
19.2 Controlling Access to Areas within Live Documents	290
19.2.1 Scenario of Multiple End Users with Different Access	291
19.2.2 Overview of the Process Used to Control Access to Objects	292
19.2.3 Design Considerations for Using Design Groups to Control Editing	296
19.3 Using DLF Keys to Provide Temporary Editing Access to Live Documents	297
19.3.1 Types of DLF Keys	298
19.3.2 Setting Up DLF Keys	299
19.3.3 Applying DLF Keys to Live Documents	301

Chapter 20: Implementing Security Features in Live Documents	303
20.1 Encrypting Data and Content in a Live Document	303
20.1.1 Live Encryption Switches	304
20.2 Preventing Content from Appearing in Thumbnail Previews of Live Document Files	307
20.3 Protecting Variable Data by Using Privacy Masks	308
20.3.1 About Privacy Masks and End User Editing	308
20.3.2 Using Privacy Masks with Output Formatting	309
20.3.3 About Privacy Masks and Live Document Data	309
20.3.4 Setting Up Privacy Masks	310
20.4 Using Signature Buttons to Allow Certain End Users to Prevent Changes in Live Documents	311
20.4.1 Adding a Signature Button to a Page	311
20.4.2 Defining the Variable Settings of the Signature Button	312
20.4.3 Customizing the Appearance of the Signature Button	313
20.4.4 Selecting the Scope of the Signature	315
20.4.5 Specifying Additional Signature Button Actions	316
20.4.6 Preventing End Users From Removing a Signature	317
Chapter 21: Testing Live Documents before Moving into Production	318
21.1 Using Live Preview in Designer to Test Interactive Components	319
21.2 Previewing Output Colors in a Live Document	321
21.2.1 About Previewing Output Colors in LiveEditor	321
21.2.2 Adding Colors to a Color Data File	322
21.2.3 Removing Preview Colors from Your Color Data File	323
21.2.4 Loading a Color Data File	324
21.3 Locating Live Objects on a Page	324
21.4 Using the Debugger to Test Logic in a Live Document	325
21.4.1 The Debugger at a Glance	325
21.4.2 Controlling How Results Appear	327
21.4.3 Controlling the Level of Detail Recorded for Rules, Formulas, and Functions	331
21.4.4 Testing Specific Objects in a Live Document	332
21.4.5 Viewing the Current Value of Variables	343
21.4.6 Viewing the Design Properties of Objects	345
21.4.7 Sharing Debugger Results	346
Chapter 22: Generating Live Documents	348

22.1 Including the Appropriate Resources in a Package File	348
22.1.1 Determining Which Resources to Include in a Package File	349
22.1.2 Specifying Which Resources to Include in a Package File	349
22.2 Special Timing Considerations for Live Documents	350
22.2.1 Setting the Default Variable Substitution and Rule Execution Time	351
22.2.2 Using Rules to Specify Execution Timing and Control Object Inclusion ..	352
22.2.3 Overriding the Default Variable Substitution and Rule Execution Time for Specific Objects	354
22.3 Setting Up the Objects Necessary for Live Output	354
22.3.1 Creating a Live Setting Object	355
22.3.2 Creating a DLF Output Object	355
22.3.3 Creating a DLF Output Queue	357
22.3.4 Defining How Resources Are Handled in the Output	358
Chapter 23: Processing Live Documents	360
23.1 Using One-to-One Processing for Live Documents	360
23.1.1 Using One Live Document as a Driver File	362
23.1.2 Using Multiple Live Documents as Driver Files	363
23.1.3 Setting Up an Application to Use One Live Document or Multiple Live Documents as Driver Files	364
23.1.4 Importing a Live Document at Run Time	371
23.2 Using One-to-Many Processing for Live Documents (Live Documents as Templates)	371
23.2.1 Using a Driver File Maintained Outside of the Live Document	373
23.2.2 Using a Distribution List Contained in a Live Document	376
23.3 Using Live Engine Switches During Processing	380
Appendix A: Live Built-In Functions	381
A.1 LiveAction	382
A.2 LiveBalloon	383
A.3 LiveBeep	384
A.4 LiveCheck	385
A.5 LiveClose	386
A.6 LiveDataFileClose	388
A.7 LiveDataFileOpen	388
A.8 LiveDebugMessage	390
A.9 LiveEnvironment	391

A.10	LiveEraseSignature	393
A.11	LiveExternalDocImport	394
A.12	LiveGetSignature	395
A.13	LiveGetSignatureByIndex	397
A.14	LiveGetSignatureCount	398
A.15	LiveGoTo	399
A.16	LiveLaunchURL	401
A.17	LiveMessage	403
A.18	LiveNavigate	405
A.19	LiveOpen	406
A.20	LivePrint	407
A.21	LiveSave	408
A.22	LiveSaveAs	409
A.23	LiveSaveAsPDF	410
A.24	LiveSelection	413
A.25	LiveSendEmail	414
A.26	LiveSetPrivacyMask	416
A.27	LiveSetReadOnly	417
A.28	LiveSetStyleSheet	418
A.29	LiveSign	419
A.30	LiveStore	420
A.31	LiveSubmit	421
A.32	LiveTestMembership	421
A.33	LiveValidate	422
A.34	LiveView	423
A.35	LiveXMLRead	424
A.36	LiveXMLWrite	425
	Appendix B: Live System Variables	426
B.1	Comprehensive List of Live System Variables	426
B.2	Unique Behaviors of Date System Variables in LiveEditor	428
B.3	Unique Behaviors of Page Numbering System Variables in Live Documents	429

Appendix C: LiveEditor and Live Engine Switches	432
C.1 LiveEditor Switches	432
C.1.1 ACCEPT_ALL	433
C.1.2 ACTION	434
C.1.3 DDAFILEMAP	434
C.1.4 DLFUSER and DLFPASSWORD	435
C.1.5 DSNMAP	435
C.1.6 FILEMAP	436
C.1.7 INITVARSET	436
C.1.8 PRODUCTLEVEL	437
C.1.9 WEBSERVICEMAP	437
C.2 Live Processing Switches	438
C.2.1 DISABLE_EMBED_DATA	439
C.2.2 DISABLE_FLOW_TARGETS_FOR_LIVE	439
C.2.3 DLFCAMPDISABLE	439
C.2.4 DRIVERLISTFILE	440
C.2.5 PROMOTECAMPTODLF	440
C.2.6 RETAIN_VARIABLE_RESET_TIME	441
C.2.7 SHOWREVISION	441
C.2.8 USE_DLF_STYLE	442
C.3 Live Encryption Switches	442
C.3.1 DLF_CIPHER	442
C.3.2 DLF_CIPHERLIST	443
C.3.3 DLF_CIPHERLOG	444
C.3.4 DLF_CIPHEROPTIONS	444
C.3.5 DLF_CIPHEROUT	445

Chapter 1: Getting Started Designing Interactive Documents

- “About Interactive Documents” below
- “How Interactive Documents Work with Exstream” on page 16
- “Creating Your First Live Document Application” on page 22
- “A Technical Look at the DLF File Structure” on page 27

1.1 About Interactive Documents

Not every document that you produce with Exstream goes directly to the printer. Human interaction is an important step in many enterprise customer communication processes. Whether it's collecting information face-to-face with a client, generating correspondence while on the phone in a call center, or presenting data collection forms to customers on a company website, Exstream enables businesses to streamline the business processes that support customer interactions and helps you communicate more effectively with your customer base.

By using Exstream Live, you can produce interactive documents that extend the power of Exstream and allow you to eliminate cumbersome manual document processes that are not integrated with back office workflows. Interactive documents are designed and created in the same environment as other Exstream applications. Unlike traditional Exstream output, however, interactive documents are documents with which end users can "interact"—that is to say, they can make choices that cause the document to be populated with variable data, or trigger additional document composition processes. This document describes how you can create, deploy, and manage interactive documents using Exstream Live.

1.1.1 What Interactive Documents Can Do For Your Business

The Exstream Live solution provides a way for more users to contribute to the document creation and data collection processes, while reducing the need to maintain and enforce the use of templates for various communications. By eliminating point solutions for field-delivered documents—from proposals, letters, contracts, and forms, to other types of communications completed interactively—you can reduce costs and improve consistency, resulting in more productive employees, significantly reduced costs, less strain on IT resources, fewer calls to customer service, and happier customers. Because the Exstream Live solution fits seamlessly into the larger Exstream solution, you can leverage the familiar and powerful design and

processing features in Exstream Design and Production while creating interactive document solutions that bring new flexibility and efficiency to your organization.

1.1.2 Examples of Live Documents You Can Produce

Any business process that involves end users interacting with documents, or any interactive document process that exists to spawn other processes or generate additional documents, can potentially be a requirement you meet with interactive documents. Some of the application types particularly suited for the Exstream Live solution include the following:

- Claims
- Collateral fulfillment
- Contracts
- Correspondence
- Direct mail
- Enrollment/account opening kits
- Group policies
- HR forms
- Proposals

1.1.3 About LiveEditor and Empower Editor

Exstream Live includes two interactive document solutions:

- **LiveEditor**—LiveEditor is a stand-alone program, available with Exstream version 6.0 and later. LiveEditor can either be installed on end users' computers or accessed through Internet Explorer (using a browser add-on). You create Live documents in Exstream Design and Production for end users to edit in LiveEditor, and you use a traditional production workflow to fulfill Live documents that have been edited by end users.
- **Empower Editor**—Empower Editor is a thin-client solution, available with Exstream. Empower Editor is installed on an application server for end users to access through an HTML 5-compliant web browser. You create Empower documents in Exstream Design and Production for end users to edit in Empower Editor, for a controlled editing experience. Once the user has completed the desired changes, you can incorporate the Empower document into a traditional production workflow, and use the Exstream production engine to fulfill the document.

For Exstream version 16.6.0, the information in this guide addresses designing Live documents and interactive document workflows that use LiveEditor. However, many of the included

processes also apply to designing Empower documents and interactive document workflows that use Empower Editor. The guide *Designing for Exstream Empower Editor* in the Exstream Design and Production documentation, details the differences between Live document workflows and Empower document workflows. For Exstream version 16.6.0, you will need to use both of these guides when developing applications that create or fulfill Empower documents.

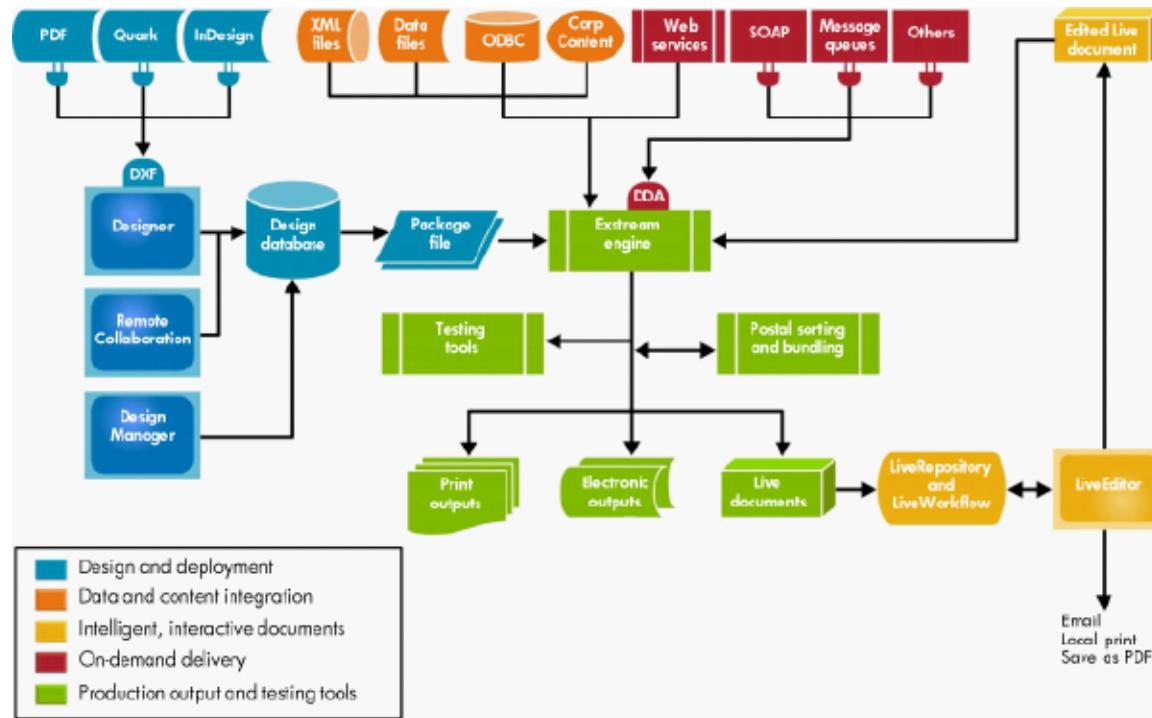
1.2 How Interactive Documents Work with Exstream

Interactive documents extend the power of Exstream with functionality-rich templates that eliminate cumbersome manual document processes. You use the powerful design and management tools of Exstream Design and Production to create interactive documents, and then use the engine to create an interactive document that will be distributed as dictated by your workflows. Edited interactive documents can be sent back to the Exstream engine to drive other processes, such as automating fulfillment; updating corporate systems, records management, and archive systems; or making copies of the edited document in other formats.

In the following illustration, notice how the Live document functionality of Exstream Live fits into the larger Exstream architecture and provides additional options for delivery to customers:

- A package file is produced and run through the engine to produce Live documents.
- The resulting Live documents might be incorporated into a Live workflow or stored within a Live repository for subsequent editing using the LiveEditor.
- End users can interact with the Live document by using the LiveEditor in stand-alone mode, or through a web browser, such as Internet Explorer.
- After edits are complete, the Live document can be emailed, printed locally, or saved to PDF from within the LiveEditor. Or, the end user can submit the edited Live document back to the engine so that a different output type, such as AFP, is produced.

Exstream architecture



The process for creating and managing Empower documents is similar to that for Live Documents. The main differences are that end users use Empower Editor in a web browser for editing documents, the documents are fulfilled using Exstream Engine as a Web Service (EWS), and you set up the document workflow in the Empower Services Layer.

For more information about using EWS, see *Configuring Connectors* in the Exstream Design and Production documentation.

For more information about creating Empower documents and managing your workflow when using Exstream Empower Editor, see *Designing for Exstream Empower Editor* in the Exstream Design and Production documentation.

1.2.1 The Design Environment

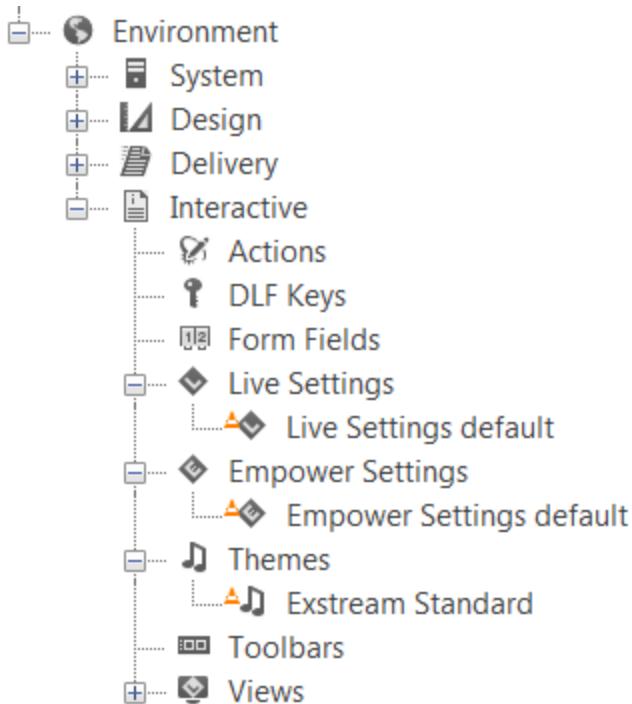
To create an interactive document, you use the familiar design and administration tools in the Exstream Design and Production environment. The process of designing an interactive document is similar to designing traditional types of Exstream documents. You use the design and management tools in Design Manager to create the building blocks of the application, including objects such as pages, documents, data files, and variables. You then use the design tools in Designer to design pages that contain the content, data, and objects that make up the document.

However, with Exstream Live, you can take advantage of additional design properties. Most often, you access these properties on the **Interactive** tab of the object properties. Different

fields are available on the **Interactive** tab, depending on the type of object you are setting up. For example, if you are creating a text box or table cell that you want to allow end users to edit, you can use the **Interactive** tab of the object properties to define the editing permissions for that object.

In addition to the **Interactive** tab, additional headings are available in the Library, under the **Interactive** heading, to support the management of the interactive document.

Interactive heading and subheadings in the Library



1.2.2 The Editing Environment

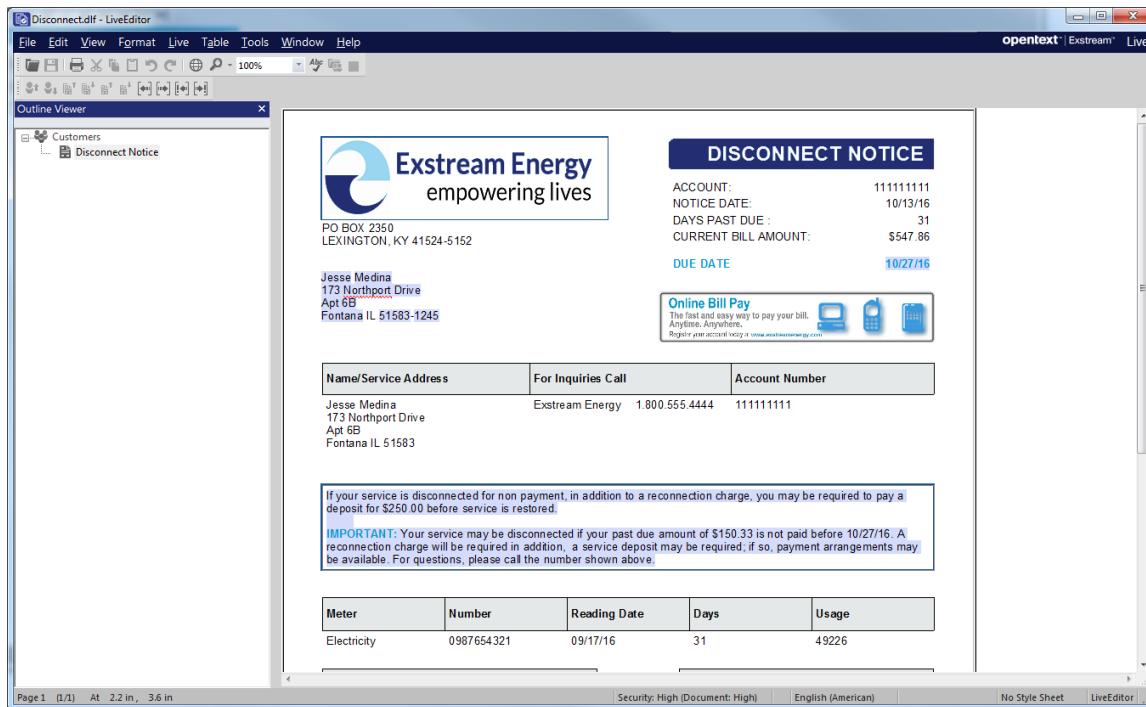
Typically, the editing of, and interaction with, interactive documents takes place in LiveEditor or Empower Editor. LiveEditor is a stand-alone program that can either be installed on end users' computers or accessed through Internet Explorer. Empower Editor is a thin client solution that an end user accesses through a web browser. These interactive editors are similar to Designer in that they allow end users to perform many of the same tasks. They each have a more simplified interface, however, and not all of the functionality that is available in Designer is available in LiveEditor or Empower Editor, since the purpose of the interactive editor is to edit and interact with interactive documents, not create them. The editing capabilities available to end users can be further enhanced or restricted by the way you design the interactive document and the permissions you give each end user.

For more information on controlling how an end user can edit a Live document, see “[Controlling End User Editing Using Selection Controls](#)” on page [76](#).

For more information about creating documents for Empower Editor, see *Designing for Exstream Empower Editor* in the Exstream Design and Production documentation.

The following graphic shows the LiveEditor interface with a Live document open for editing.

Example of a Live document open in LiveEditor for editing



Usually, editing a Live document in LiveEditor is simply one step in the document life cycle. However, depending on your organization's workflow, you might choose to leverage the editing phase differently. For example, if you send a Live document to regional offices for customization, those end users might complete the Live document, print it locally, and then send it directly to customers. In this instance, the editing environment might be the final stage of the document life cycle. On the other hand, some organizations might choose not to use LiveEditor at all in some document processes. For example, you can design a Live document that is transformed into a web form. The document is never opened in LiveEditor; instead, its data is edited online through a webpage. After that, the data is reunited with the Live document and sent to the engine for processing.

1.2.3 The Processing Environment

After an interactive document has gone through the workflow that you designed for it, you can return it to the Exstream Design and Production engine for processing. One of the advantages of the Exstream Live solution is that you can take advantage of many of the processing features and benefits available with Exstream Design and Production. For example, you might use dynamic content import to include custom content in a Live document during fulfillment.

In addition to importing an interactive document as-is to a variety of output types, you can also access the data contained within the interactive document and use it to drive other document processes. For example, a Live document can be designed with an interview page used to collect data about a new customer. When the Live document is returned to the engine, the data drives the assembly of a new customer packet, which is personalized with the customer's unique information.

1.2.4 Live and Empower Modules Available from Exstream

The following modules enable the Exstream interactive capabilities:

Live modules

Module name	Description
LiveOutput (DLF Output)	<p>LiveOutput lets you produce DLF files, or Live documents. Live documents are security-enabled and based on WinZip and XML standards and they require LiveEditor or LiveViewer to be opened, viewed, and edited.</p> <p>In addition to the ability to produce Live documents, the LiveOutput module also includes the functionality of the following modules:</p> <ul style="list-style-type: none">• Web Services Interface (in order to make web service calls from within LiveEditor)• XML/JSON Input (limited functionality to enable mapping of auxiliary layout files) <p>When you license the LiveOutput module, you do not have to purchase the Web Services Interface and XML/JSON Input modules separately.</p>
LiveEditor	<p>LiveEditor is a desktop tool that is separate from Exstream. It provides a controlled, interactive user experience for DLF files. LiveEditor can be opened as a stand-alone application or within Microsoft Internet Explorer. LiveEditor will conform to the DLF file that it opens to enforce permissions, editing, data and content access, and printing. LiveEditor is licensed by seat (or user).</p> <p>LiveEditor is typically not included in configurations that leverage Live Certificates unless there is a need to limit the ability to edit a Live document (DLF) after a certain date (enforce DLF expiration). Only Windows platforms are supported, and browser compatibility is limited to Microsoft Internet Explorer.</p>
LiveViewer	<p>LiveViewer lets you view and print a read-only version of a DLF, unless the DLF file contains a LiveCertificate (.dkf file) that grants additional permissions. LiveViewer is available free with the purchase of a Live module.</p>

Live modules, continued

Module name	Description
Live Certificates	<p>Live Certificates let you configure a DLF so that it can be edited by end users of LiveEditor, as well as end users of LiveViewer. This allows you to distribute a DLF to a wide audience and provide them with limited editing capabilities in the free LiveViewer. Live Certificates offer an alternative licensing option to LiveEditor that is based on potential user count. Live Certificates are purchased in tiers and are meant to reflect the number of potential users who will, at some time, receive a DLF to edit. For example, a 5,000 Live Certificate tier allows up to 5,000 potential users to edit an unlimited number of Live documents. (The certificate count—5,000—is meant to be an indication of potential users, not the total number of DLF files that can be edited.) Customers receive a key (.dkf file) to load into their design database. This key is then, in turn, inserted into Live documents. The key allows LiveViewer to behave as LiveEditor when a document that contains a certificate (.dkf file) is opened with LiveViewer.</p> <p>This licensing method is based on a "potential user" model rather than a "concurrent user" model. This type of implementation is recommended for large-volume applications, such as call centers and line of business enablement applications.</p> <p>The only functional difference between Live Certificates and LiveEditor is that Live Certificates automatically allows for expiration dates for a Live document. Live Certificates are not metered within the engine, but are contractually enforced.</p>
InteractiveInput	InteractiveInput lets you use DLF files as data files to drive data and content extraction, trigger events, or create other documents. This module adds an input type to the design environment that allows a Live document (DLF file) to be sent directly to the engine as a driver file, thus enabling subsequent composition events for alternative outputs or additional documents. It also lets you use DLF files as content to be included in another document and delivered in any format. This capability is typically required for most fulfillment applications where customization has occurred within the LiveEditor.
InteractiveFulfillment	InteractiveFulfillment lets you use a DLF file as a template in place of a customer driver file during on-demand or high-volume production. When they are used as a template, DLFs can support applications for documents that must be edited before printing, such as direct mail offerings, sales brochures, and form letters. When the DLF is reintroduced to the engine as a template, the engine populates the variables and triggers the rules in the DLF, and completes fulfillment processing by producing individual documents for each customer in the run. InteractiveFulfillment is needed to upload customer information (driver files) from within LiveEditor.
LiveRepository	LiveRepository is server-based and provides the content storage and management that is necessary for building a Live application. A Java-based component, it provides storage, backup, and access rights to DLF files.
LiveWorkflow	LiveWorkflow is server-based and provides file orchestration within a given workflow or review process. A Java-based component, it sits on top of the repository to model the process steps and execute them within a workflow.
Live Application Library	Live Application Library (LAL) is software development kit that is designed to speed deployment of Exstream interactive solutions. LAL provides the most common modules and connectors that are required to integrate Exstream into your existing IT and content environment. It is licensed as a server-based component, and a separate license is necessary for LiveRepository and LiveWorkflow installation. LAL is highly recommended for all deployments of LiveRepository and LiveWorkflow.

Empower modules

Module	Description
Empower Output	<p>The Empower Output module is required to produce files that support the Empower document capabilities of Exstream. Empower documents are used to present and collect data and content from Empower Editor end users.</p> <p>This module requires the following additional modules:</p> <ul style="list-style-type: none">• Web Services Interface• InteractiveInput (requires the Dynamic Content Import module)• PDF Output
Empower Server	<p>The Empower Server hosts the Empower Administrative Console and the Empower Services Layer. The Empower Services Layer uses REST APIs to communicate with the Empower Editor to provide interactive pages to end users. The Empower Editor supports the following API calls:</p> <ul style="list-style-type: none">• Open document• Export document• Import document
Empower Editor	<p>The Empower Editor provides editing capability for interactive documents directly through a web browser. The Empower Editor eliminates the need to deploy and install software on individual devices. If you are an Exstream Design and Production customer, you can use existing content from your Exstream Live applications with Empower. Empower provides a subset of the functionality provided in LiveEditor.</p>

1.3 Creating Your First Live Document Application

Since the process of creating a Live document application can require new and additional considerations beyond those used in designing a traditional Exstream application, it might be helpful to review this section before creating an application.

This section discusses the following topics:

- [“Overview of the Live Document Application Development Process” on the next page](#)
- [“Application Planning Considerations for Creating a Live Document” on page 24](#)

1.3.1 Overview of the Live Document Application Development Process

Since the process of creating a Live document application is basically the same as the process of creating a traditional Exstream Design and Production document, you can follow the same basic application development process for both. For example, the same roles and responsibilities you assign for traditional applications can still be helpful during the development process for Live document applications.

For information about the typical application design process, see *Designing Customer Communications* in the Exstream Design and Production documentation.

However, there are some notable differences and additions to keep in mind when you design a Live document:

- **Different audiences**—Depending on your workflow, you might be designing a single document that will be used by both the editing audience and your final customer. Make sure that your design makes it easy for end users to know how they must interact with the Live document and what they must change.
- **Additional testing time**—You must allow for additional time and testing of the interactive properties of the document you create, as well as its final output appearance, since you must verify the document's appearance in both the editing phase and the final production phase.
- **Different data considerations**—Not only must data be available during the initial engine run when the Live document is created, but it might also need to be available when the Live document is edited and during a final engine run.
- **Additional design objects and properties**—When you set up a Live application, you use Live-specific objects and properties to define the behavior of the Live document. Most objects are optional, and you use them only as needed for a particular Live document. However, a setting object is always required in order to create Live output. A setting object acts as a container object for other Live-specific objects and properties, making it easy to manage those objects and properties in one place.

For more information about Live-specific design objects and properties, see “[The Design Environment](#)” on page 17.

For information about creating a setting object, see “[Creating a Live Setting Object](#)” on page 355.

1.3.2 Application Planning Considerations for Creating a Live Document

The following sections discuss a variety of considerations to keep in mind when you plan your Live document. These considerations include the following:

- “[Consider the Design](#)” below
- “[Consider the Editing Experience](#)” on the next page
- “[Consider the Data and Logic](#)” on page 26
- “[Consider the Resources](#)” on page 27

Consider the Design

As you design a Live document, it is important to consider the life cycle of the document. For example, if the Live document will be used primarily to collect data that will drive other processes, you might want to focus your design efforts on making the interactive experience consistent and easy to use. On the other hand, if a Live document, such as a proposal or contract, will be edited by end users and then sent to customers, you might choose to focus on using the Exstream design tools effectively to create a document with a professional, clean appearance.

Design planning questions

Planning question	Recommendation
Are the users who will be contributing to the Live document experienced with the document before it was interactive?	<p>The design tools in Designer give you the flexibility to design any type of document, such as proposals, forms, or statements, so you can recreate legacy documents or create documents that are similar to old ones. Consider whether it will be helpful to structure the document similarly to the previous version, or if you should use conventions that will help ease the transition to the new document. Feature such as themes and pop-up help tips can help end users become acquainted with working in new documents.</p> <p>For information about using Live document features to assist end users with editing a Live document, see “Using Live Tools to Create a More Intuitive and Automated Editing Experience” on page 123.</p>

Design planning questions, continued

Planning question	Recommendation
Should the document design change to incorporate those desired improvements that have not been a priority previously?	<p>The transition to a Live documents solution can be an ideal time to make changes or improvements to a document's design. For example, consider whether there is an issue in the current design that you have never had the time or commitment from other groups to address. With Designer, you can easily implement the change, and then make adjustments to it as needed.</p> <p>Similarly, consider whether there were guidelines in previous document templates that end users followed but were not enforced. You can easily design a Live document that enforces the guidelines you want all documents to follow, such as limiting the formatting users can apply to text.</p> <p>For information about controlling the ways different end users can edit a Live document, see "Controlling the Formatting Options Available to End Users" on page 247.</p>
Is accessibility and/or Section 508 compliance required?	If accessibility and/or Section 508 compliance is required, consider how those requirements will affect the design before beginning the document creation process.

Consider the Editing Experience

The Exstream Design and Production environment provides some specific features you can use when creating a Live document to control the editing experience.

Editing experience planning questions

Planning question	Recommendation
Does the Live document contain sensitive information that requires security measures during the document life cycle?	<p>Live documents provide several types of security features you can use to protect sensitive customer information or to limit which end users access the documents.</p> <p>For more information about the security features you can use in a Live document, see "Implementing Security Features in Live Documents" on page 303.</p>
Will multiple users contribute to the editing process?	<p>If so, do you need to "lock down" editing of certain areas to certain users? You can use design groups to control which end users have access to specific areas of a Live document.</p> <p>For information about controlling the ways different end users can edit a Live document, see "Controlling Access to Live Documents and Areas within Live Documents" on page 286.</p>
What types of editing or interaction will be required for the end users?	<p>Depending on the ways end users will be interacting with a Live document, you can employ several types of features to help guide them to make the proper types of changes to the document. For example, you can use views and themes to provide visual guidance to end users. You can also create tooltips and pop-up help that gives end users guidance on the type of changes that must be made in a specific area.</p> <p>For information about using Live document features to assist end users with editing a Live document, see "Using Tools for Providing Guidance to End Users" on page 217.</p>

Consider the Data and Logic

Review the following questions and incorporate the recommendations into your application planning as needed:

Data and logic planning questions

Planning question	Recommendation
When during the Live document life cycle should variable values and rules be executed?	<p>In traditional applications, variables and rules are executed when the document is generated by the engine. However, with Live documents, variable values might need to be updated and rules might need to be executed throughout the documents' life cycle. For example, you might need to run a rule during editing, based on new information provided by the end user.</p> <p>By default, variable values are set to be updated and rules are set to be executed during the initial engine run only. Keep in mind that if you need to change this behavior, you can set the default behavior that applies to all variables and rules, and the behavior for specific variables and rules.</p> <p>For information about setting execution times, see "Setting the Default Variable Substitution and Rule Execution Time" on page 351.</p>
What editing activities should be automated?	<p>There are many ways to automate processes that will occur during the editing phase. Automating processes can help prevent errors from occurring, speed up the editing tasks, and can provide an easier, more friendly editing environment for end users.</p> <p>Live documents allows you to execute many tasks and processes automatically through the use of actions. Actions associate functions with a Live document so that end users can execute them.</p> <p>Live documents also provides many built-in functions that are specific to Live document processes.</p> <p>For more information about automated editing activities, see "Using Live Tools to Create a More Intuitive and Automated Editing Experience" on page 123.</p>
How will the data collected during editing be used?	<p>If you will use the data provided when the Live document is edited to drive other processes, you must often make accommodations for that during the creation of the Live document. For example, if you will use the data entered in a Live document to drive the creation of other documents during a fulfillment process, you must account for that use of the data during the initial design decisions.</p> <p>For more information about setting up fulfillment processes, see "Processing Live Documents" on page 360.</p>
Do you want to allow end users to add variables to the Live document?	<p>If so, make sure you carefully communicate to end users about the result of adding variables. By default, no variables are set up so that they can be added to a Live document; if you want to enable this functionality, you must set up variables to be included in the Live document.</p> <p>For more information about setting up variables so that end users can add them to a document, see "Setting Up a Variable Palette to Allow End Users to Insert Variables" on page 119.</p>

Consider the Resources

Review the following questions and incorporate the recommendations into your application planning as needed:

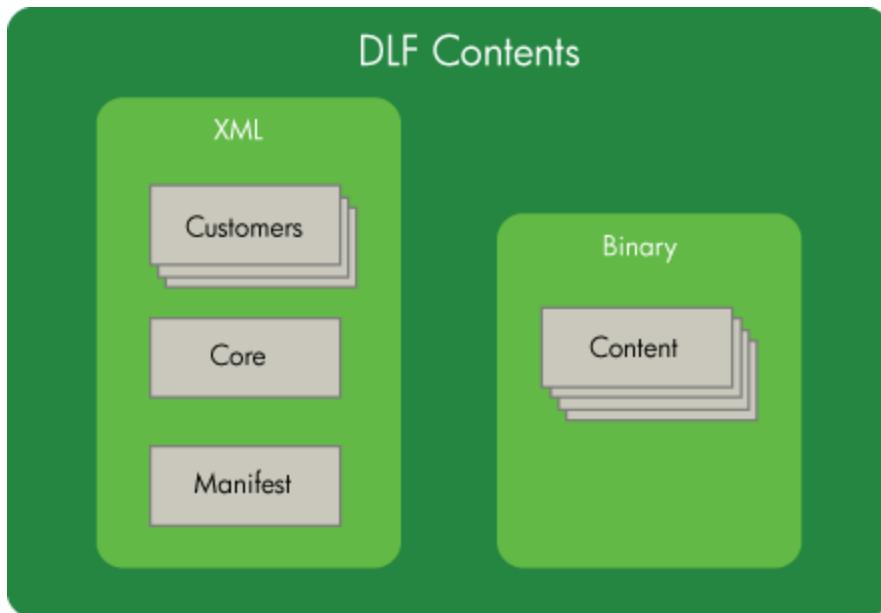
Design planning questions

Planning question	Recommendation
What types of fonts do end users need to access?	Because the visual appearance of a Live document can be changed after its initial design, you can control how fonts are used when text is formatted. When setting up the Live document, make sure that you include the fonts that are required for end users to give it the required appearance. On the other hand, you can also ensure that end users cannot use fonts that are not approved for use in corporate communications. For more information about controlling access to fonts, see "Controlling Font Behavior in Live Applications" on page 248 .
Will end users be allowed to create PDFs of the Live document?	If you allow end users to save Live documents as PDFs, you can control how some of the PDFs resources are handled when the PDF is created. For more information about allowing end users to create PDFs, see "Controlling End User Options for Saving a Live Document as a PDF" on page 262 .

1.4 A Technical Look at the DLF File Structure

One of the greatest strengths of Exstream Live documents is the underlying file structure that comprises the documents. Live document files have a DLF extension. DLF is a compressed file format that is organized by an internal folder structure. The data and content that make up a DLF are stored as binary and XML files within this folder structure. The XML files contain open and accessible areas containing data and metadata in well-formed XML format, which allows you to parse and manipulate the XML data using third-party software or scripts. In addition, the consistent format makes it easy to organize and search a DLF repository for specific data.

Data and content storage in a Live document



Because DLFs can contain sensitive customer data, Live documents provide several options for protecting the data within DLFs and the DLFs themselves.

For more information about the security features Live documents provide, see [“Implementing Security Features in Live Documents” on page 303](#).

Note: DLF output is Open Packaging Convention (OPC)-compliant.

This section discusses the following topics:

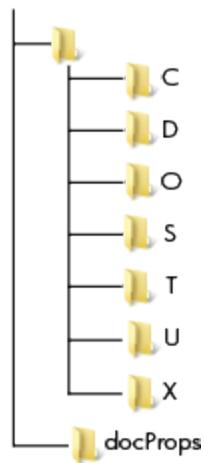
- [“The DLF Folder Structure” on the next page](#)
- [“Viewing and Accessing the Structure of a Live Document” on page 31](#)

1.4.1 The DLF Folder Structure

The data and content that make up a Live document are stored in various folders within the DLF file. The following illustration shows how the folders are used to organize the data and content within the DLF file.

Illustrated DLF folder structure

Live document



Some of these folders are not present when the Live document is initially created and are added later in the workflow as the Live document changes. The folders contain the following types of content:

- **DLF**—Data, content, and objects
 - C—Composed pages of output
 - D—Variable customer data
 - O—Design objects such as images and fonts
 - S—Debug information
 - T—Customer data, such as driver files
 - U—End-user data, such as electronic signatures and uploaded data files
 - X—Data from interview pages that can be displayed as web forms
- **docProps**—Metadata about the document

The following table describes the most common types of information you will need to access in a Live document and where they are located within the DLF folder structure.

Live document information and location

Type of information	Detail	Located in folder
Customer data	<p>One XML file is created for each customer and contains the variable data for that customer. The XML files follow the numbering scheme of <code>Customer1.xml</code>, <code>Customer2.xml</code>, and so on, rather than following the numbering scheme of the customer driver file. As end users interact with a customer's information in the Live document, any changes made to the data are stored in the customer's XML file.</p> <p>Note: Since each XML file is customer-specific, there is no default schema.</p>	D
Live document metadata	Metadata about the Live document itself is stored in the <code>DialogueLiveCore.xml</code> file. You can use the information contained in this file to determine when the Live document was created, the application name, the current version of the Live document, and other information.	docProps
DLF file properties	A list of the Live document's properties is stored in the <code>Manifest.xml</code> file. This file contains a list of each item in the Live document (such as customers, documents, and pages) and the properties of each.	C
Revision history	If you select User selectable or Always on as the Revision tracking method on the Live setting object, one <code>RevHistoryLog.xml</code> file is created for the entire Live document. However, if you select User selectable , keep in mind that the end users must also turn on revision tracking in LiveEditor in order for the <code>RevHistoryLog.xml</code> file to be created. If revision tracking is turned on in LiveEditor, the <code>RevHistoryLog.xml</code> file contains complete information about tracked revisions for each customer. The <code>RevHistoryLog.xml</code> is identical to the XML file that you create on the History tab of the Revision Viewer Palette in LiveEditor.	C
Revision history of variables	If you select Deltas or Full as the XML history tracking method on the Live setting object, files for each customer are placed in a <code>Revision<#></code> folder. The XML files in the <code>Revision<#></code> folder contain only information about changes to variables within the Live document.	D
Interview page data	<p>If you select the Use as a Web form check box on the properties of an interview page, the data and objects on pages are included in XML format within the Live document.</p> <p>For more information about using an interview page as a web form, see "Creating an Interview Page to Collect Data" on page 198.</p>	X

Note: Double-byte character set content is encoded in the UTF-16 little-endian format.

1.4.2 Viewing and Accessing the Structure of a Live Document

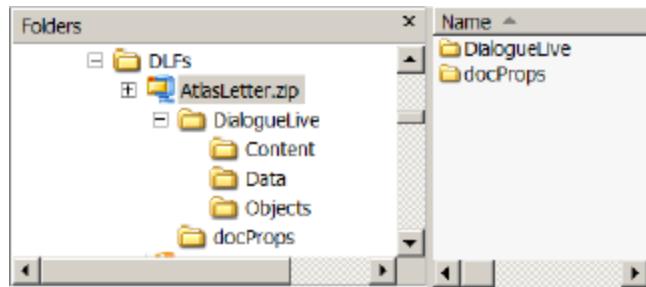
Viewing the structure of a Live document lets you see the data contained within the XML files and view the structure of the compressed files. To view the folder structure and the contents of a DLF, you use a ZIP program (such as WinZip) and file explorer program. If you access the data contained within a Live document during your document life cycle, you can use this process to acquaint yourself with how data is stored within the Live document.

To view the structure of a Live document:

1. Save the Live document with a .zip extension. For example, rename NameOfFile.dlf to NameOfFile.zip.
2. Right-click the file and select **Explore**.

The folder opens in a window and you can view the folder structure.

View of Live document folder structure



Chapter 2: Managing Live Document Versions in an Enterprise Environment

If you use Live documents in an enterprise environment, you most likely create Live documents that will be distributed to many different end users and then often returned to Exstream for processing. In a complex enterprise environment, it can be a challenge to manage the distribution of Live documents when end users have different versions of LiveEditor and LiveViewer. However, Exstream Live provides several features you can use to help manage Live documents in your enterprise environment. You can use the following features in Exstream Live to help manage enterprise implementation requirements:

Features for managing Live document versions

I want to	Feature
Allow end users to edit Live documents in older versions of LiveEditor and LiveViewer	<p>You can select the Allow Live document usage check box to allow newer versions of a Live document to be used with older versions of LiveEditor and LiveViewer. Because this setting might cause compatibility issues (depending on the Live features included in your design), you should review the information about Live versions before using it.</p> <p>You can also use the custom message feature to communicate to end users that a new version of LiveEditor is available, or you can suppress messages completely. When you use a custom message, you can also provide a custom hyperlink that end users can click to get more information or to access a newer version of LiveEditor or LiveViewer.</p>
Require end users to upgrade	<p>You can clear the Allow Live document usage check box to prevent newer versions of a Live document from being used with older versions of LiveEditor and LiveViewer. An end user who tries to open the Live document in an older version of LiveEditor or LiveViewer will receive an error message, and the Live document will not be opened.</p>

This chapter discusses the following topics:

- “[Overview of Live Document Versioning](#)” below
- “[Allowing End Users to Edit Live Documents in Older Versions of LiveEditor](#)” on page 35
- “[Requiring End Users to Upgrade](#)” on page 36

2.1 Overview of Live Document Versioning

Each Live document has an internal version number, which is assigned when the Live document is generated. The version of the engine (and therefore, the version of Exstream) determines the version of the Live document, and each version of Exstream creates a different Live document version number. For example, Exstream 7.0 produces Live documents at version 2.0, and Exstream 7.0.1 produces Live documents at version 2.0.1. Live document version numbers are

used to ensure that Live documents are compatible with the version of the LiveViewer or LiveEditor being used to open them.

As a Live document travels through your organization's workflow, its version number can change multiple times. A Live document's version number changes when the Live document is saved in a LiveEditor version that is newer or older than the version that corresponds with the engine version with which the Live document was originally created. When an end user saves a Live document, the Live document takes on the version number of the LiveEditor version in which the document is saved. Versions of LiveViewer or LiveEditor that are older than a Live document might not support all of the functionality available in that Live document. For example, some of the features in a Live document at version 2.0 might not be fully supported in LiveEditor version 1.0.

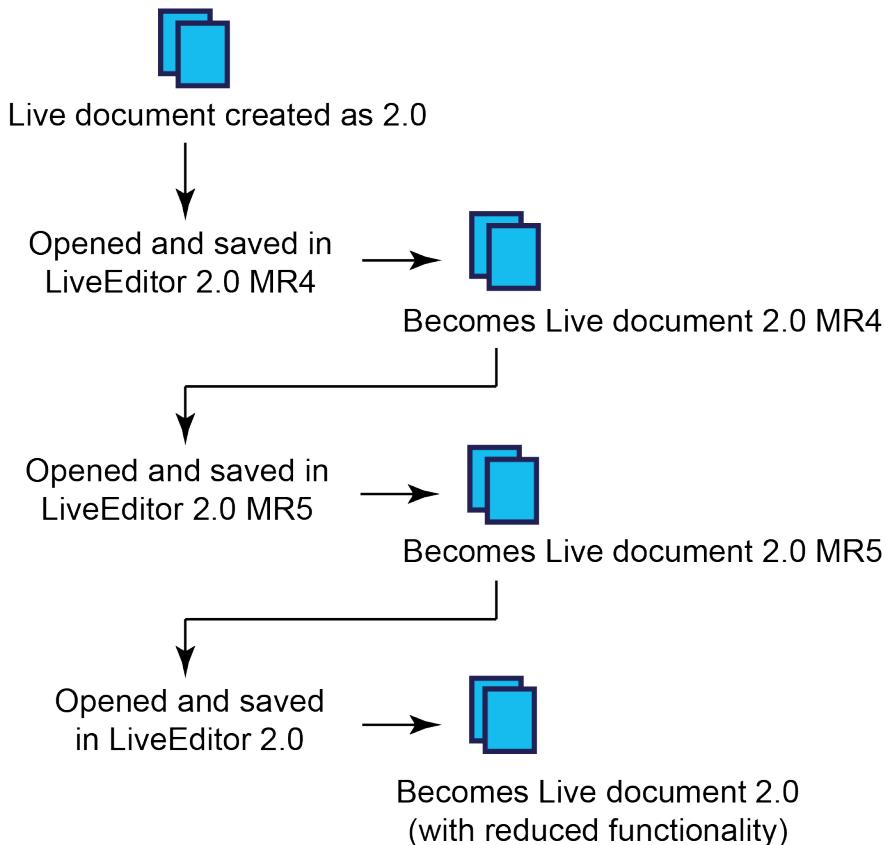
When a Live document changes versions, the functionality in the document is affected. The following table describes how versions in a Live document can affect the editing process and workflow of the document:

Versioning behavior and considerations

If the Live document is	Then
Edited and saved in an older version of LiveEditor	<p>Any functionality not supported in that version of LiveEditor is lost. For example, suppose a Live document is created at version 3.0. At some point during the workflow, an end user using LiveEditor version 2.0 opens and saves the Live document. At that point, any functionality that is supported only in version 3.0 is lost. Even if the Live document is reopened and resaved in a LiveEditor version 3.0, the functionality cannot be reinstated.</p> <p>If a Live document will be processed in a fulfillment process, it is recommended that you not allow the Live document to be edited and saved in older versions of LiveEditor. When Live documents are saved in older versions of LiveEditor, supporting information in the documents are lost, which can cause them to fail during fulfillment.</p>
Opened and edited in a newer version of LiveEditor	All of the functionality included in the original Live document design is available and can be used by the LiveEditor. However, newer features in the LiveEditor might not be supported for that Live document.

In the following illustration, the Live document was created as version 2.0. However, as it passes through the workflow and is opened and saved in various versions of LiveEditor, the version of the Live document changes, based on the versions of LiveEditor that are used.

Example of Live document version changes



A Live document's version number also changes if the Live document is repackaged with a newer version of the engine than the engine version with which it was originally created. In that case, the Live document takes the version number of the LiveEditor version that corresponds to the version of Exstream used to package it.

Tip: You can find information about the current Live document version in the `DialogueCore.xml` file within the Live document.

2.2 Allowing End Users to Edit Live Documents in Older Versions of LiveEditor

In certain circumstances, you can allow end users to edit newer Live documents with older versions of LiveEditor. However, before you enable this behavior, review how Live documents change versions and the potential LiveEditor of allowing end users- editing to change the Live document version.

You must also make sure that the packaging and database serialization version numbers for the instance of Design Manager that you use to create the Live document are not higher than the packaging and database serialization version numbers for the instance of LiveEditor that end users use to open it. For example, suppose the instance of Design Manager that you use to create a Live document uses packaging and database serialization versions of 800 and 850, respectively. If the end user's instance of LiveEditor uses packaging and database serialization versions of 795 and 845, then that instance of LiveEditor displays a serialization error and will not open the Live document.

You can use the following methods to find the serialization versions for Design Manager and LiveEditor on the information panel of each program:

- In Design Manager, from the Menu bar, select **Help > About Design Manager**.
- In LiveEditor, from the Menu bar, select **Help > About**.

You can also include a custom message that is issued when end users attempt to open a Live document in an older version of LiveEditor or LiveViewer. For example, you might create a message that says, "To ensure compatibility, please download the latest version of LiveEditor before continuing." In addition to the custom message, you can also provide a hyperlink that end users can click to access an internal IT site or My Support. Alternatively, you can suppress the warning messages that appear when the documents are opened or saved in LiveEditor or LiveViewer. Keep in mind that it is typically a best practice to allow these warnings, rather than suppress them, because they inform end users that functionality in the Live document that is not supported by that version of LiveEditor or LiveViewer will be lost when opening or saving the file. However, you might want to suppress these warnings when you have designed the Live document so that no issues will occur if the end user edits it in an earlier LiveEditor version.

To allow end users to edit Live documents in older versions of LiveEditor:

1. In Design Manager, drag the setting object associated with the Live document to the Property Panel.

The setting object opens for you to define.
2. Click the **Editor Framework** tab.
3. In the **Usage with older LiveEditor versions** area, select the **Allow Live document usage** check box.

4. To suppress or customize the messages that end users receive when they open the Live document in an older version of LiveViewer, or when they open or save the Live document in an older version of LiveEditor, complete one or more of the following tasks:

To	Do this
Suppress the default warning message that end users receive when they open a newer Live document in an older version of LiveEditor or LiveViewer	In the SUPPRESS COMPATIBILITY WARNING area, select the ON OPENING check box.
Suppress the default warning message that end users receive when they save a newer Live document in an older version of LiveEditor	In the SUPPRESS COMPATIBILITY WARNING area, select the ON SAVING check box.
Define a custom message and, optionally, a link that appears when an end user opens the Live document in an older version of LiveEditor or LiveViewer	<ol style="list-style-type: none">a. From the SUPPRESS COMPATIBILITY WARNING options, clear the ON OPENING check box.b. In the CUSTOM MESSAGE TO USE IN LIVEDITOR box or the CUSTOM MESSAGE TO USE IN LIVEVIEWER box, enter the message that you want to appear when an end user opens the Live document in the respective program.c. If you want to provide a link to a website where end users can get more information or access a newer version of the program, enter the URL in the LINK TO URL box. For example, you might enter http://www.opentext.com.d. If you want to provide text that appears in place of the hyperlink text, enter the text in the TEXT TO DISPLAY box. For example, suppose that you provide a hyperlink to http://www.opentext.com. Then, you might enter Visit the OpenText Web site in the TEXT TO DISPLAY box so that the text appears instead of the hyperlink, while still providing a link to the site.

5. From the Menu bar, select **File > Save**.

2.3 Requiring End Users to Upgrade

There might be times when you want to prevent end users from opening and saving Live documents in older versions of LiveEditor to ensure that no functionality is lost during the editing process, or to prevent end users from opening Live documents in older versions of LiveViewer because newer functionality might not work correctly. In these cases, you can require end users to upgrade before they can open a DLF by disabling their ability to open the Live document in an older version of LiveEditor or LiveViewer. An end user who tries to open the Live document in an older version of LiveEditor or LiveViewer will receive an error message, and the Live document will not be opened.

To require end users to upgrade:

1. In Design Manager, drag the setting object associated with the Live document to the Property Panel.
The setting object opens for you to define.
2. Click the **Editor Framework** tab.
3. In the **Usage with older LiveEditor versions** area, clear the **Live document usage** check box.
4. From the Menu bar, select **File > Save**.

Chapter 3: Creating a Basic Live Document

After planning the Live document that you want to create and creating the basic components of the document, you must make the appropriate objects "Live." For example, if you have decided that you want end users to customize a text paragraph, then after creating the paragraph, you must define its Live properties so that it is editable by end users. This chapter discusses how to make most of the basic Exstream design objects Live and provide some simple editing capabilities to your end users. This chapter also discusses special considerations for using certain design objects in a Live document, including how to set up a Live document to dynamically import text and images at run time.

After you have created a basic Live document, you can then use the other information in this guide to add more robust and automated editing capabilities to the Live document.

This chapter discusses the following topics:

- ["Allowing End Users to Change Text" on the next page](#)
- ["Allowing End Users to Change the Value of a Variable" on page 45](#)
- ["Allowing End Users to Make Changes to a Table" on page 48](#)
- ["Allowing End Users to Add Images to a Page" on page 54](#)
- ["Allowing End Users to Format Chart Text" on page 56](#)
- ["Allowing End Users to Change the Size, Position, or Rotation of Objects" on page 57](#)
- ["Allowing End Users to Reorder Documents Using the Outline Viewer" on page 58](#)
- ["Defining Sections and Paragraphs for Live" on page 59](#)
- ["Setting Up a Live Document to Dynamically Import Text and Images at Run Time" on page 60](#)
- ["Using Flow Frames in Live Documents" on page 65](#)
- ["Using Tables of Contents and Indexes in Live Documents" on page 66](#)
- ["Designing Live Documents for Multiple Languages" on page 68](#)
- ["Defining Layouts for Electronic Devices That Can Be Delivered from Live Documents" on page 72](#)

3.1 Allowing End Users to Change Text

Live documents give end users the ability to change text that is located in text boxes, paragraphs, and table cells. This capability provides some of the simplest yet most powerful editing abilities available in Live documents. For example, you can easily make a text box Live by allowing end users to edit the text in it. You can also restrict text editing to smaller areas, such as a single paragraph within a text box.

You can allow end users to change the text in a Live document in three basic ways:

- Changing the content of text
- Changing text formatting
- Changing text properties (such as spacing or margins)

You can allow end users to change a combination of these settings to enable a very specific editing capability in a Live document. For example, you might want to allow end users to change text formatting but not the content of text so that an invoice will match that region's branding standards. On the other hand, you might want to allow end users to completely customize a letter to a new customer; in that case, you can allow them to change the text itself, as well as its formatting and properties.

This section discusses the following topics:

- [“Allowing End Users to Make Free-Form Changes to Text” below](#)
- [“Making Free-Form Changes to Text: The End User Experience” on page 41](#)
- [“Allowing End Users to Change Text Formatting” on page 42](#)
- [“Allowing End Users to Change Text Properties” on page 43](#)
- [“Changing Text Properties: The End User Experience” on page 44](#)

3.1.1 Allowing End Users to Make Free-Form Changes to Text

One of the most common ways to add basic editability to a Live document is to allow end users to make free-form changes to text in a text box or table cell. For example, you might choose to make the closing paragraph of a letter template editable so that end users can add a personal note to the recipient. This section discusses how to make static text in a text box, text paragraph, or table cell editable.

Note: If you want end users to edit text in a cell within an automated table, you must use variables to contain the text so that end users' changes are retained.

For an overview of how the end users will make free-form changes to text, see “[Making Free-Form Changes to Text: The End User Experience](#)” on the next page.

If you want to exert more control over the types of changes that end users can make to text, or to limit the types of text that are accepted in a specific area, Exstream Live provides several different features you can use to control that type of editing. For example, if end users must enter a Social Security number, you can use a form field to ensure that end users enter the appropriate number of digits. If you need this level of control over the way end users can change text, you should use one of the form control objects available in Live documents, rather than allowing end users to make free-form changes to text.

For information about controlling the types of changes end users can make to text, see “[Controlling the Formatting Options Available to End Users](#)” on page 247.

If the text consists of section and paragraph objects, keep in mind that if an end user deletes a paragraph from a section, LiveEditor does not automatically delete the corresponding entry from a table of contents. If you want to allow the end user to remove paragraphs and have LiveEditor update a table of contents automatically, you can set up a content selection group that includes the paragraphs.

For more information about using sections and paragraphs and adding a table of contents, see *Designing Customer Communications* in the Exstream Design and Production documentation.

For more information about using a content selection group, see “[Using a Content Selection Group to Dynamically Select Content](#)” on page 100.

To allow end users to make free-form changes to text:

1. In Designer, do one of the following to select the text that you want to allow end users to change:

To	Do this
Allow end users to make changes to all of the text in a paragraph	<ol style="list-style-type: none">Select the text box.Click . <p>The Text Properties dialog box opens.</p>
Allow end users to make changes to specific lines of text	<ol style="list-style-type: none">Select the lines or paragraphs of text.Right-click the selected text and select Text paragraph properties. <p>The Text paragraph properties dialog box opens.</p>

2. Click the **Interactive** tab.
3. From the **Content change** drop down list, select either **Change is optional** or **Change is**

required, as needed.

Additional options become available.

4. Select the **Text edit** check box.
5. From the adjacent drop-down list, select **Standard text**.

Tip: Alternately, you can select **Data field**. The **Data field** option enables the same functionality; however, when an end user initially places the cursor in an area labeled as a data field, the entire editable area is highlighted. If an end user places the cursor in an area identified as a **Standard text** editable area, the area is not highlighted. Instead the cursor begins blinking to indicate that the end user can begin editing the text.

6. Click **OK**.
- The dialog box closes.
7. From the Menu bar, select **Edit > Save**.

3.1.2 Making Free-Form Changes to Text: The End User Experience

This section describes how end users will interact with and edit text that you allow them to change in LiveEditor.

1. In a Live document, end users will select the text they want to change. Text that can be changed will be indicated by the theme you design.

Editable text in LiveEditor

Dear John,

Let us plan your next vacation! Whether you are looking for a romantic getaway, relaxing, or exciting adventures or family fun, we can make it happen. We can make your airline reservations departing from any airport of your choice to any part of the world. We can arrange hotel packages to any destination or advise you on suitable accommodations for you, your family, or your group.

Lost paradises, romantic getaways, ancient haciendas, elegant villas, intimate boutique hotels, tantalizing spas, private departures, personalized service, traditional cuisine, original itineraries, and fabulous activities and tours. These are the ingredients of our very special Romantic Adventures Collection, and exclusive concoction of romance and soft adventure, where days are spent negotiating rapids, observing wildlife, kayaking in pristine lakes, horseback riding or snorkeling in crystalline waters and evenings are filled with the sight of star-studded skies, the scents of homemade recipes and the sounds of romance.

As a Worldwide Travel client, you may select from several special travel packages which have been developed specifically for you. If you would like more information about these great getaways, contact Client Services at 512-603-7400.

2. End users can enter new text or edit the existing text as needed.

3.1.3 Allowing End Users to Change Text Formatting

You can allow end users to make changes to the formatting of text, such as its font, style, and size. For example, if the Live document will be sent to regional offices that each use unique branding standards, you can allow end users to change the text formatting so that the correspondence for each region uses the appropriate font and color.

Tip: You can use style sheets and the **Allowed fonts** property on the setting object to control the types of fonts and font styles that can be used in the Live document.

For information about controlling the types of fonts and font styles that end users can use, see “[Controlling the Formatting Options Available to End Users](#)” on page 247.

To allow end users to change text formatting:

1. In Designer, do one of the following to select the text that you want to allow end users to format:

To	Do this
Allow end users to format of all the text within a text box or table	<ol style="list-style-type: none">Select the text box.Click . <p>The Text Properties dialog box opens.</p>

To	Do this
Allow end users to format only a specific paragraph of text in a text box or table cell	<ol style="list-style-type: none">a. Select the lines or paragraphs of text.b. Right-click the selected text and select Text paragraph properties. <p>The Text paragraph properties dialog box opens.</p>

2. Click the **Interactive** tab.
3. From the **Content change** drop down list, select either **Change is optional** or **Change is required**, as needed.
Additional options become available.
4. Select the **Text format** check box.
5. Click **OK**.
The **Text paragraph properties** dialog box closes.
6. From the Menu bar, select **Edit > Save**.

3.1.4 Allowing End Users to Change Text Properties

If end users will make extensive changes to text, you might want to allow them to change the text properties so that they can adjust the appearance of the text after their changes are made. Giving end users the ability to change text properties allows them to customize the text in a Live document more extensively than simply changing the text formatting. For example, using the text properties, end users can easily add a bulleted or numbered list to the text.

When you allow end users to change text properties, they have access to a subset of the text properties available in Designer. Keep in mind, however, that many of the properties that end users have access to might be unfamiliar to them. If you choose to allow end users to change text properties, make sure that you provide information about the properties with which end users are not familiar.

For an overview of how the end users will change text properties, see “[Changing Text Properties: The End User Experience](#)” on the next page.

There are two ways you can allow end users to change text properties:

- **Change the text properties of a specific text paragraph in a table cell or text box**—Allows end users to make adjustments such as the line spacing and paragraph breaking properties.
- **Change the text properties of a text box**—Allows end users to make adjustments such as the text margins, text border, and text autofit.

You might need to enable property editing for both text boxes and text paragraphs to provide the level of flexibility end users need. Test your design to determine which type of property editing is best for your design.

Note: If you allow end users to change the properties of a text box, the column properties will be inactive in LiveEditor. Therefore, end users will not be able to change the numbers of columns in the text box.

To allow end users to change text formatting and properties:

1. In Designer, do one of the following to select the text for which you want to allow end users to change properties:

To	Do this
Allow end users to change the properties of a text box	<ol style="list-style-type: none">Select the text box.Click . <p>The Text Properties dialog box opens.</p>
Allow end users to change the properties of a text paragraph	<ol style="list-style-type: none">Select the lines or paragraphs of text.Right-click the selected text and select Text paragraph properties. <p>The Text paragraph properties dialog box opens.</p>

2. Click the **Interactive** tab.
3. From the **Content change** drop down list, select either **Change is optional** or **Change is required**, as needed.

Additional options become available.
4. Select the **Text properties** check box.
5. Click **OK**.

The **Text paragraph properties** dialog box closes.
6. From the Menu bar, select **Edit > Save**.

3.1.5 Changing Text Properties: The End User Experience

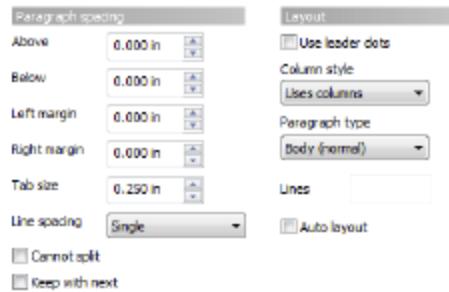
This section describes how end users will change text properties in LiveEditor.

1. In a Live document, end users will select the text whose properties they want to change. They can either select a text paragraph, or, if they can change the properties of an entire

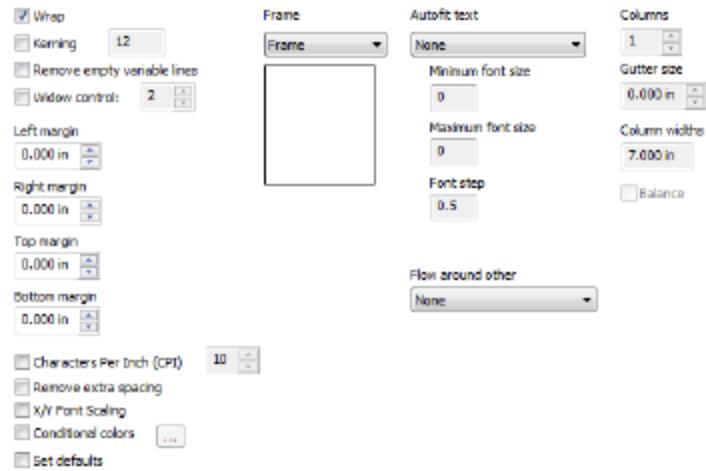
text box, they can place the cursor in the text box.

2. By right-clicking and using the shortcut menu, end users will open the properties dialog box for either the text paragraph or text box dialog box.

Properties that can be changed for a text paragraph



Properties that can be changed for a text box



3. End users will make changes to the properties as needed, and close the dialog box when they are finished.

3.2 Allowing End Users to Change the Value of a Variable

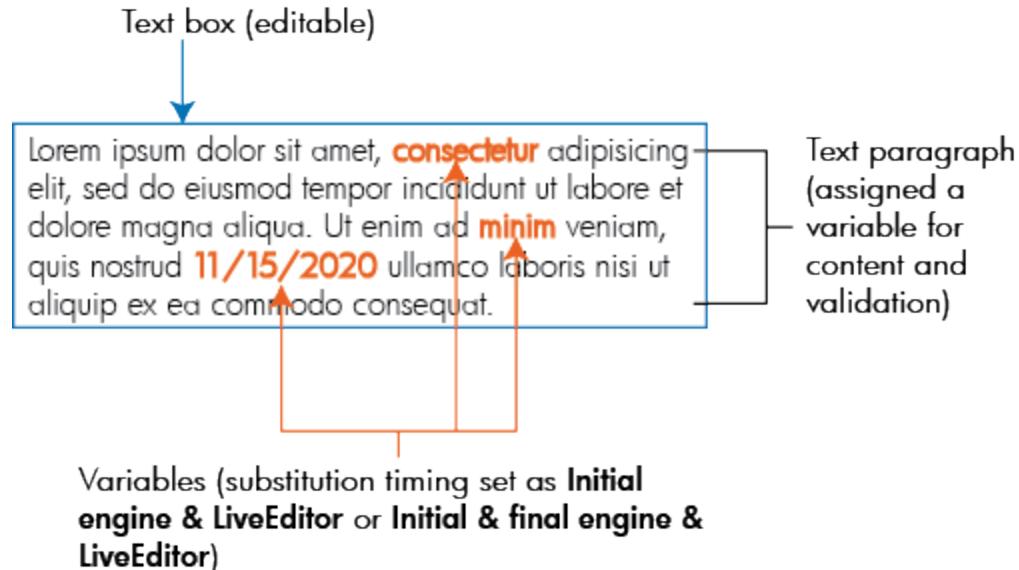
Since you can leverage variables in so many ways in the Exstream environment, allowing end users to change the value of a variable can be a very powerful feature of a Live document. For example, suppose you use Live documents to create enrollment documents. You can use

editable variables to collect data about a new enrollee, such as his or her name, address, and date of birth. Because the enrollee's information is stored in variables, it can be used to drive other processes, such as a fulfillment process that creates a custom mailing for the individual, or it can be retrieved and used to update internal systems, such as a records management system. If you will allow end users to change the value of a variable, you might want to consider using one of the form controls available in Live documents to exert more control over the types of data an end user can enter. For example, if end users will update the 'Customer_Phone_Number' variable, you can use a form field to ensure that end users enter the appropriate number of digits. If you need this level of control over the way end users can change text, you should use one of the form control objects available in Live documents, rather than allowing end users to make free-form changes to variables.

For information about controlling the types of changes end users can make to text, see ["Controlling the Formatting Options Available to End Users" on page 247](#).

When creating Live documents, you will avoid problems and confusion if you do not place editable variables within editable text. If an editable variable is contained within a section of editable text, it is not only difficult for end users to determine whether they are making changes to the static text or to the variable, it can also produce unexpected results in LiveEditor. The following example illustrates a design that would both be confusing to end users and produce unexpected results in LiveEditor:

Editable variables in an editable text paragraph



In this example, because of the substitution timing of the three variables within the text box, each variable is independently editable. Furthermore, because the paragraph has been assigned a variable for content and validation, the design embeds three variables within another variable.

To avoid problems in this design, you can set the substitution timing for the three variables in the paragraph to **Initial engine run only**. This change will make the variables behave like the rest of the editable text in the paragraph. As a result of the timing change, however, if end users make updates to the variable text, the updated values will not be reflected anywhere else that

the variable is used in the Live document. As an alternative, you can place the editable variables between interactive areas, but doing so would require end users to click each interactive area to make changes (and would result in seven separate interactive areas in the design shown in the example).

Keep in mind that you can make system variables editable only if the **can be set** check box is selected on the **Basic** tab of the system variable properties.

For information about advanced design considerations when using variables in Live documents, see [“Setting Up a Variable Palette to Allow End Users to Insert Variables” on page 119](#).

Variables used on interview pages (sometimes called “interview variables”) behave differently from other types of variables in Exstream. Because interview pages are meant to collect data that will affect the rest of the document, the values of variables used on interview pages persist throughout the pages in the document. Therefore, keep in mind that if you allow end users to change the value of interview variables, those changes will appear throughout the document (in each place the variable appears).

For more information about interview pages and how variables used on them behave, see [“Using Interview Functionality to Drive Data Changes” on page 123](#).

To allow end users to change the value of a variable:

1. In Designer, select the variable that you want to allow end users to change.
2. Right-click the selected variable and select **Variable Interactive area**.
The **Variable Properties** dialog box opens, with the **Interactive** tab active.
3. Click the **Interactive** tab.

Note: The **Interactive** tab is not available on the **Variable Properties** dialog box if the **can be set** check box is cleared on the system variable properties.

4. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.
Additional options become available.
5. Select the **Text edit** check box.
Additional options become available.

Tip: You can use these additional options to help control the types of data end users enter.

For information about using Live features to control editing, see [“Controlling End User Editing Using Selection Controls” on page 76](#).

6. Click **OK**.

The **Variable Properties** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

3.3 Allowing End Users to Make Changes to a Table

Tables are robust design objects that you can use to present complex information, particularly transactional or financial information. Depending on the type of table you include in your design (either simple or automated), you can enable different types of changes that can be made by the end user. For example, in simple tables, you can allow changes to be made to all of the text within a table. On the other hand, in automated tables, changes to text are controlled through individual interactive areas within the table. This section discusses how to set up simple tables for editing separately from how to set up automated tables for editing.

Note: For all types of tables used in Live documents, you can use the show/hide feature on specific rows to remove individual table rows from the table without deleting them. When a row is hidden, the row is removed from view, but the content is still available. For example, if you are creating a statement and you do not want to show transactions less than five dollars, you can hide the rows containing such transactions. Though hidden from view, the rows still contribute to the running totals in the table. The show/hide feature is available on the **Interactive** tab of row properties.

For more information about using the show/hide feature, see *Designing for LiveEditor* in the Exstream Design and Production documentation.

This section discusses the following topics:

- “Table Design Considerations for Live Documents” below
- “Allowing End Users to Edit Simple Tables” on the next page
- “Allowing End Users to Edit Automated Tables” on page 51

3.3.1 Table Design Considerations for Live Documents

You can include any table type available in Exstream Design and Production in a Live document. If you include automated rows or automated columns in a table, you can also allow end users to delete or duplicate specified rows or columns, as well as to insert new rows or columns. Keep in mind that, just as in Exstream Design and Production, you should not design a table in Exstream Live that contains both automated rows and automated columns. By the same

principle, you cannot allow end users to add, delete, or duplicate both rows and columns in the same table.

Because tables in Exstream are complex objects, it is important to understand how the choices you make as you design tables will affect the table when the Live document is edited. The tables you include in an Exstream Live application can use most of the features that are available for tables in non-Live applications. However, keep in mind the following differences as you design tables to be used in a Live document:

- If you include automated tables in your Live document and you want end users to edit text in a cell, you must use interactive variables or content variables to contain the text. If end users change plain text in any table cell, the changes are lost when the page flows or is saved
- The **Remove empty rows** feature causes rows that do not contain text or other objects to be removed during the initial engine run. Use caution if you use this feature in a Live document, since some empty rows might be needed during the fulfillment step of the Live document life cycle. Rows that are required for use in fulfillment appear when the Live document is created and are visible in LiveEditor. To hide rows so that they do not appear in LiveEditor, you can use the show/hide feature or rules to suppress rows with empty variables.
- Column rules applied to an automated column are fired only once for each column. For example, if a column repeats five times based on five elements in an array, and a rule is run that targets element one and excludes it, all five occurrences will be excluded. The rule is not executed for subsequent occurrences of the same column.
- If you design a table that includes automated columns, the table must be set to split and flow and a flow page must be included in the design.
- If you design a table that includes automated rows and you embed the table in a paragraph object, the table and the paragraph object must each be set to split and flow and a flow page must be included in the design. If you place an automated table on a page, however, do not set the table object to split and flow.
- The following features and behaviors available for tables in Exstream Design and Production are not supported in Live documents:
 - Serpentine rows
 - Growing up (tables cannot grow in an upward direction during editing)
 - Legend boxes

3.3.2 Allowing End Users to Edit Simple Tables

If you use simple tables in a Live document, you can set up tables in which end users can edit and format the text in the table. You can also give end users the ability to control the height of individual rows within the table.

This section discusses the following topics:

- “[Allowing Text in Simple Tables to Be Edited or Formatted](#)” below
- “[Allowing the Height of Rows to Be Changed](#)” below

Allowing Text in Simple Tables to Be Edited or Formatted

If you include simple tables in a Live document, you can allow end users to edit all of the content within them, rather than enabling changes for specific cells. You can allow end users to edit both the content and the formatting of content in simple tables.

To allow end users to edit simple tables:

1. In Designer, select the table that you want to allow end users to edit.

2. Click .

The **Table Properties** dialog box opens.

3. Click the **Interactive** tab.

Note: The **Interactive** tab is not available on the **Variable Properties** dialog box if the **can be set** check box is cleared on the system variable properties.

4. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.

Additional options become available.

5. Select the **Text edit** and/or **Text format** check box, as needed.

6. If you want to force end users to add a comment when they edit the table, select **Require a comment on edit** check box.

For more information on using comments in Live documents, see “[Setting Up Live Document Review Tools](#)” on page 267.

7. Click **OK**.

The **Table Properties** dialog box closes.

8. From the Menu bar, select **Edit > Save**.

Allowing the Height of Rows to Be Changed

1. Select the row you want to allow end users to change.
2. Right-click the selected row and select **Row Properties**.

The **Row Properties** dialog box opens.

3. Click the **Interactive** tab.
4. From the **Content change** drop-down list, select **Change is optional**.
5. Select the **User can adjust height** check box.
6. Click **OK**.

The **Row Properties** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

3.3.3 Allowing End Users to Edit Automated Tables

Because automated tables are designed to grow and change as data changes and logic is executed, you can allow different types of changes in automated tables than you can in simple tables. In automated tables (any table that contains automated rows), you can allow rows and columns to be added, duplicated, or removed. You can also control the appearance of sections within the tables to make the table easier to read and navigate.

This section discusses the following topics:

- “[Allowing Automated Rows to Be Added, Duplicated, or Removed](#)” below
- “[Allowing Automated Columns to Be Added, Duplicated, or Removed](#)” on the next page
- “[Expanding and Collapsing Table Sections](#)” on page 53

Allowing Automated Rows to Be Added, Duplicated, or Removed

You can allow end users to add, duplicate, or remove automated rows in tables. However, automated rows that repeat based on a fixed number cannot be added, duplicated, or removed.

To allow automated rows to be added, duplicated, or removed:

1. In Designer, depending on what action you want the end users to take, complete one of the following steps:

To	Do this
Allow end users to duplicate a row	Select the row you want end users to duplicate.
Allow end users to delete a row	Select the row you want end users to delete.

To	Do this
Allow end users to insert a new row	Select the row below where you want to allow end users to insert a new row. Note: If you let end users insert rows, the rows use the initial value of array element 1. If you do not specify an initial value, the inserted row is blank.

2. Right-click the selected row and select **Row properties**.

The **Row Properties** dialog box opens.

3. Click the **Interactive** tab.
4. From the **Content change** drop-down list, select **Change is optional**.

Additional options become available.

5. Select one or more of the check boxes to enable the type of changes you want end users to make.
6. Click **OK**.

The **Row Properties** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

Allowing Automated Columns to Be Added, Duplicated, or Removed

You can allow end users to add, duplicate, or remove automated columns in tables. However, automated columns that repeat based on a fixed number cannot be added, duplicated, or removed.

To allow automated columns to be added, duplicated, or removed:

1. In Designer, depending on what action you want end users to be able to take, complete one of the following steps

To	Do this
Allow end users to duplicate a column	Select the column you want end users to duplicate.
Allow end users to delete a column	Select the column you want end users to delete.

To	Do this
Allow end users to insert a new column	Select the column to the right of where you want to allow end users to insert a new column. Note: If you let end users insert columns, the columns use the initial value of array element 1. If you do not specify an initial value, the inserted column is blank.

2. Right-click the selected column and select **Column properties**.

The **Column Properties** dialog box opens.

3. Click the **Interactive** tab.

4. From the **Content change** drop-down list, select **Change is optional**.

Additional options become available.

5. Select one or more of the check boxes to enable the type of changes you want end users to make.

6. Click **OK**.

The **Column Properties** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

Expanding and Collapsing Table Sections

The expand/collapse feature lets you collapse a table section in the output. When a table section is collapsed, the header and footer associated with the table section remain visible to help identify what information is not visible. For example, if you are creating a year-end financial statement that has a section for debits and one for credits, you might want to collapse the sections and show summaries rather than individual transactions. In this way, you can use the transactions to create the summary, but you need not show them in the table.

For information about table sections, see *Designing Customer Communications* in the Exstream Design and Production documentation.

To expand or collapse table sections:

1. In Designer, select the header row that starts the table section you want to expand/collapse.
2. Right-click the selected row and select **Row properties**.

The **Row Properties** dialog box opens.

3. Click the **Automated Row Properties** tab.
4. From the **Row type** drop-down list, select one of the header options.
5. In the **Table Section (set of rows)** area, configure the table section.

6. Click the **Interactive** tab.
7. From the **Content change** drop down list, select **Change is optional**.
8. Select the **Expand/Collapse table section** check box.
9. In the **Expand/collapse variable** box, select the Boolean variable that indicates whether a row is expanded or collapsed for a specific table section.

a. Click .

The **Select Variable** dialog box opens.

b. Select the variable you want to use and click **OK**.

The **Select Variable** dialog box closes and the variable you selected appears in the **Expand/collapse variable** box.

10. Click **OK**.

The **Row Properties** dialog box closes.

11. Repeat step 6 through step 12 for each section you want to expand or collapse.

12. From the Menu bar, select **Edit > Save**.

When the table is composed, some sections will be collapsed based on the value of the Boolean variable selected for the table section.

3.4 Allowing End Users to Add Images to a Page

The most basic way to allow end users to customize the images in a Live document is to allow end users to add their own image files to a page. When you allow end users to add images to a page, you set up an area in which an image can be added. Then, during editing, end users can select a local image file to be placed in that area. For example, an insurance agent might upload a picture of herself to add a personal touch to a welcome letter. End users can add images only in defined areas on the page; they cannot add images if you have not set up an area that allows an image to be placed there.

Tip: Another way to allow end users to customize the images in a Live document is to create an image selector. Image selectors allow end users to select an image from a pre-set group of images that is included in the Live document.

For information about setting up an image selector, see “[Using an Image Selector to Allow End Users to Choose From a Set of Images](#)” on page 91.

The following types of images can be uploaded to a Live document in LiveEditor:

- BMP
- EMF
- EPS

Note: When an end user adds an EPS image to a Live document, vector data from the image is rasterized for display in LiveEditor. If you want end users to import EPS images, for best results, install Ghostscript on the same computer as LiveEditor. If an end user imports an image in LiveEditor without first installing Ghostscript, the image might appear distorted in LiveEditor. The original vector image data is stored in the Live document for use in fulfillment to outputs that support vector images, regardless of whether Ghostscript is installed on the computer on which the end user imports the image into LiveEditor.

Ghostscript is a commercially available PostScript and PDF conversion and rendering tool. For information about installing Ghostscript, see *Installation and Upgrade Information* in the Exstream Design and Production documentation.

- FAX format
- GIF
- JPEG
- IMG
- MAC
- MSP
- PNG
- TIFF
- WMF

Keep in mind that if end users want to import images with transparency, the images must be in PNG format. Transparency in other image formats is not supported in LiveEditor. Also keep in mind that only certain output types support transparency in PNG images included in a Live document.

For more information about the output types that support transparency in PNG images, see *Importing External Content* in the Exstream Design and Production documentation.

To allow end users to upload or paste an image to a page:

1. In Designer, on the Drawing Objects toolbar, click  .

The **Create an empty image** dialog box opens.

2. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.

Additional options become available.

3. From the **Use control to select content** drop-down list, select **File upload**. When you select **File upload**, end users can manually upload an image file. The **File upload** option also allows end users to copy and paste an image into an empty image object, or to replace an existing image with a pasted image.

End users can add an image through the paste option by right-clicking in the selected empty image object and selecting **Paste** in the context menu, by pressing CTRL + V, or from the Menu bar, by selecting **Edit > Paste**.

4. If you want to provide a tip for the end users to inform them they can upload or paste their own image, enter the tip text in the **Prompt to display when there is no image** box. For example, you might enter `Upload your own image here`.

5. Click **OK**.

The **Create an empty image** dialog box closes and the placeholder image area appears on the page.

6. From the Menu bar, select **Edit > Save**.

Note: You cannot use empty image objects with multiple-page import placeholder variables.

3.5 Allowing End Users to Format Chart Text

Because charts are entirely data-driven, end users can change only the formatting of text used on the chart. For example, suppose end users will add the body of a statement letter and can format the text to use their branch's specific branding standards. You can allow the end users to make changes to a chart illustrating the customer's investment allocations so the chart title and labeling appear in the same font style and size as the rest of the statement. You cannot control the individual areas of chart text to which end users can make changes; if you allow them to format the chart text, they can make changes to any area of text on the chart.

To allow end users to make format chart text:

1. In Designer, select the chart that you want to allow end users to format.
 2. Click .
- The **Chart Properties** dialog box opens.
3. Click the **Interactive** tab.
 4. From the **Content change** drop down list, select either **Change is optional** or **Change is**

required, as needed.

Additional options become available.

5. Select the **Text format** check box.

6. Click **OK**.

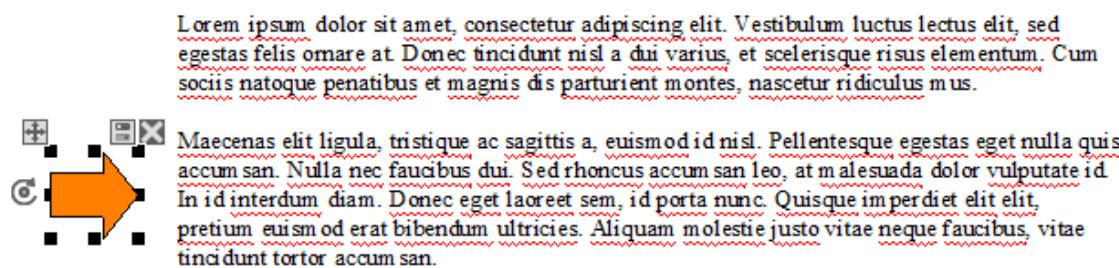
The **Chart Properties** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

3.6 Allowing End Users to Change the Size, Position, or Rotation of Objects

Although LiveEditor is not designed to be a document design tool, you can allow end users to make basic changes to the appearance of objects using LiveEditor. For example, if you use an arrow shape to draw attention to a specific area of text in a proposal, you can allow end users to change the position of the arrow in case the emphasized text moves to a different line. If end users can enter free-form text, you might also want to allow them to resize text boxes to accommodate text that becomes too long for the original text box size.

Arrow that can be resized, moved, and rotated in LiveEditor



Praesent euismod mollis aliquet. Nulla et cursus nunc. Nunc et iaculis ligula. Aliquam erat volutpat. Proin et magna quis arcu vestibulum facilisis sagittis at ante. Donec porttitor neque in nibh aliquam, eu congue tellus viverra. Quisque semper, justo non ultrices tempus, risus orci aliquet dolor, vel rhoncus lacus magna eu eros.

End users cannot change the size, position, and rotation of all types of objects. For example, end users can rotate only shapes and text boxes but no other object types. End users also cannot move automated tables or move any type of object off the page.

Tip: If your business processes require that end users be able to contribute to the physical design of document, consider using other Exstream solutions to enable this type of workflow.

To allow end users to change the size, position, or rotation of objects:

1. In Designer, select the object that you want to allow end users to resize, move, or rotate.
2. Click .
- The **<Object> Properties** dialog box opens.
3. Click the **Interactive** tab.
4. In the **Object placement** area, select the **Size**, **Move**, and **Rotate** check boxes as needed to enable the type of editing required for the object.
5. Click **OK**.
- The **<Object> Properties** dialog box closes.

6. From the Menu bar, select **Edit > Save**.

3.7 Allowing End Users to Reorder Documents Using the Outline Viewer

By default, end users cannot reorder documents using the LiveEditor Outline Viewer. Depending on your design, however, you might want to give them that ability. For example, if you allow end users to import external DOCX, DXF, PDF, or TIFF files into a placeholder document, they might need to rearrange documents into a more logical order. Keep in mind that section-driven documents and XML node-driven documents cannot be moved using the Outline Viewer panel in LiveEditor, because the underlying data determines the order in which these documents appear.

For more information about importing external files into placeholder documents, see “[Importing Pages from External Files into a Live Document](#)” on page 166.

To allow end users to reorder documents using the Outline Viewer:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Authorization** tab.
3. Select the **Allow documents to be re-ordered in the Outline Viewer** check box.
4. From the Menu bar, select **File > Save**.

3.8 Defining Sections and Paragraphs for Live

You use the **Interactive** tab of a section or paragraph object to determine how end users interact with the object or how the object appears when it is included in a Live document. For example, you can use the tab to add a description that appears when an end user hovers over the object in the Outline Viewer panel so that the object is easier to identify.

When you are using sections and paragraphs in a Live document, keep in mind that if paragraphs are listed in a table of contents, and you allow end users to make free-form changes to the text in the paragraphs, then LiveEditor does not delete the entry for a paragraph from the table of contents if an end user deletes it from the section. If you want to allow the end user to remove paragraphs and have LiveEditor update the table of contents automatically, you can set up a content selection group that includes the paragraphs.

For more information about adding a table of contents, see *Designing Customer Communications* in the Exstream Design and Production documentation.

For more information about allowing end users to make free-form changes to text, see “[Allowing End Users to Make Free-Form Changes to Text](#)” on page 39.

For more information about using a content selection group, see “[Using a Content Selection Group to Dynamically Select Content](#)” on page 100.

The following section discusses the Live properties that are specific to section and paragraph objects: property inheritance settings.

3.8.1 Setting Up Section and Paragraph Property Inheritance

When a section or paragraph object is nested under another section to create a hierarchy, you can use the **Honor parent's properties** setting on the **Interactive** tab to control how the section or paragraph object uses the properties set on the section object at the immediate higher level (known as the parent section) in the hierarchy.

To set up section or paragraph property inheritance:

1. In Design Manager, from the Library, drag the section or paragraph for which you want to define property inheritance to the Property Panel.
2. Click the **Interactive** tab.
3. From the **Honor parent's properties** drop-down list, select one of the following options to control how the section or paragraph object inherits the properties set on the parent section:

To	Do this
Inherit all parent properties from the Content change area and from all the properties in the Advanced Properties dialog box except for Name	Select All .
For child properties that have not been changed from the default setting, inherit parent properties from the Content change area and from all the properties in the Advanced Properties dialog box except for Name	Select Inherit . For any child object property that has been changed from the default setting, that child setting will be used. For example, suppose that in the parent object properties Static is selected from the Link drop-down list and a URL is specified for the link. If in the child object properties None is selected from the Link drop-down list (which is the default setting for Link), the child inherits the parent's Link setting and the associated URL. However, if in the child object Static is selected from the Link drop-down list (which is not the default setting for Link) and a URL is entered, the static URL set for the child object is used.
Do not inherit parent properties; use all child properties from the Content change area and the Advanced Properties dialog box	Select None .

For more information about the **Content change** area for sections and paragraphs, see [“Allowing End Users to Change Text” on page 39](#).

4. From the Menu bar, select **File > Save**.

3.9 Setting Up a Live Document to Dynamically Import Text and Images at Run Time

If you have licensed the Dynamic Content Import module, you can import textual or graphic content into a Live document when you create the DLF file. For example, suppose you maintain images in an external repository. Rather than importing them as static images into a document at design time, you can use a placeholder variable and import the images during the engine run. When you use a placeholder variable to import an image, the image is updated in every location where you use the placeholder variable.

You can choose to maintain the original image size or resize the image at run time. If you choose to resize the image, you can also choose to scale or skew the image, or to maintain the original image aspect ratio, when fitting the image within the dimensions of the image placeholder. You can also choose to change the dimensions of the image placeholder to match the size of the image.

For more information about maintaining the original image size or resizing images that are imported at run time, see *Importing External Content* in the Exstream Design and Production documentation.

Keep in mind that the information discussed in this section pertains only to dynamic import during the initial engine run, not dynamic content import during the LiveEditor session or during fulfillment.

For more information about allowing end users to import images during a LiveEditor session, see [“Dynamically Importing Images From a Shared Location” on page 95](#).

For more information about dynamically importing external DOCX, DXF, PDF, and TIFF files into placeholder documents, see [“Importing Pages from External Files into a Live Document” on page 166](#).

You can allow end users to change the images that were dynamically imported during the initial engine run. During the initial engine run, the images mapped to the placeholder variable are placed on the page. In LiveEditor, end users can use an image selector or upload their own images to replace the imported image. For example, suppose you are creating a real estate brochure. You can use the same placeholder variable for an image of the house four times within your document and then allow end users to upload an image on the first empty image object. Each time an end user uploads a new image of the house, the image updates in every placeholder variable location.

Caution: If you have existing design pages that you are preparing for a Live document, you must convert all inline image placeholder variables (placeholder variables not referenced by an empty image object) to empty image objects. Exstream Live does not support inline image placeholder variables.

For information about converting inline image placeholder variables to empty image objects, see [“Converting Inline Image Placeholder Variables to Empty Image Objects” on page 64](#).

The following file formats can be dynamically imported at run time:

File format	Live document considerations
DXF	If you want imported text and objects to be editable in LiveEditor, you must set the default variable substitution and rule execution time to Initial engine run only using the Live setting object. For more information about setting the default variable substitution and rule execution time, see “Setting the Default Variable Substitution and Rule Execution Time” on page 351 .
EPS	When you dynamically import an EPS image in a Live document, vector data from the image is rasterized. The rasterized version of the image is then stored in the Live document and displayed in LiveEditor. If fulfillment of the Live document includes outputs that support vector images, and you want to include vector images in those outputs, you must include the image resources for that output in the package file. Otherwise, the rasterized version of the image will be used in all outputs. For more information about including image resources in a package file, see “Setting Up Revision Commenting to Capture Explanations for Changes” on page 271 .
JPEG	None

File format	Live document considerations
PDF	None
Plain text	If you want the imported text to be editable in LiveEditor, you must set the default variable substitution and rule execution time to Initial engine run only using the Live setting object. For more information about setting the default variable substitution and rule execution time, see " Setting the Default Variable Substitution and Rule Execution Time " on page 351.
PNG	Only certain output types support transparency in PNG images included in a Live document.
RTF text	If you want the imported text to be editable in LiveEditor, you must set the default variable substitution and rule execution time to Initial engine run only using the Live setting object. For more information about setting the default variable substitution and rule execution time, see " Setting the Default Variable Substitution and Rule Execution Time " on page 351.
TIFF (B&W G4 or uncompressed)	None
TIFF (Color)	None
Word (*.docx)	<ul style="list-style-type: none">If you want the imported text to be editable in LiveEditor, you must set the default variable substitution and rule execution time to Initial engine run only using the Live setting object. For more information about setting the default variable substitution and rule execution time, see "Setting the Default Variable Substitution and Rule Execution Time" on page 351.When importing DOCX files dynamically in a Live document, any portion of text formatted as hidden in the DOCX file is converted to a comment in the Live document, including the formatting. This feature lets you preserve instructions that are already included in a DOCX document for end users who enter information. For more information about using comments in a Live document, see "Setting Up Live Document Review Tools" on page 267. Keep in mind that very long comments might be truncated when displayed within the design. End users can use the Revision Viewer palette to view the entirety of long comments. For more information about using the Revision Viewer palette in LiveEditor, see "Adding, Editing, Viewing, and Deleting Revision Comments in the Revision Viewer Palette" on page 280.

Keep in mind when importing images dynamically in a Live document that PNG is the only raster format in which transparency is supported. Also keep in mind that only certain output types support transparency in PNG images included in a Live document.

For more information about importing PNG images that include transparency, see *Importing External Content* in the Exstream Design and Production documentation.

Additional considerations might apply for dynamically importing some types of text and images at run time.

For more information about dynamically importing text and images at run time, see *Importing External Content* in the Exstream Design and Production documentation.

This section discusses the following topics:

- “[Setting Up a Page to Dynamically Import Content](#)” below
- “[Converting Inline Image Placeholder Variables to Empty Image Objects](#)” on the next page

3.9.1 Setting Up a Page to Dynamically Import Content

Before you can set up a page to dynamically import content, you must first create a placeholder variable and map it to the content that you want to import.

For more information about creating and mapping placeholder variables, see *Importing External Content* in the Exstream Design and Production documentation.

To set up a page to dynamically import content during the initial engine run:

1. In Designer, on the **Drawing Objects** toolbar, click .

The **Create an empty image** dialog box opens.

2. In the **Placeholder variable** box, do the following to select the placeholder variable that identifies the image to import at run time:

- a. Click .

The **Select Variable** dialog box opens.

- b. Navigate to the variable you want to use, select it, and click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Placeholder variable** box.

3. If you want to allow end users to change the image in LiveEditor, complete the following steps (otherwise, skip to step 4):

- a. From the **Content change** drop-down list, select **Change is optional** or **Change is required**, depending on your requirements for this image. The visual indicators (themes) that you specified for objects on which changes are required or optional are used to indicate to the end user whether the image must be changed.

For more information about themes, see “[Using Themes to Visually Guide End Users](#)” on page 218.

- b. From the **Use control to select content** drop-down list, select the method by which you want end users to add images to the Live document:

- **Image selector**—End users customize the Live document with images you provide.

- **File upload**—End users customize the Live document with images they upload. When you select this option, end users can also copy and paste their own image into an empty image object with a placeholder variable.

Note: In order to view a pasted image in LiveEditor or LiveViewer, you must include the placeholder variable in the content XML of the Live document when your application meets all of the following conditions:

- The Live document is set as a template for fulfillment processing or section data processing.
- The Live document includes an empty image design object which references a placeholder variable.
- The Live output uses a data file for content.

- c. If you selected **File upload** and you want to change the text that appears until the end user uploads content, enter the new text in the **Prompt to display when there is no image** box.
 4. Click **OK**.
- The **Create an empty image** dialog box closes and an empty image box appears on the page in Designer as a placeholder.
5. Define the properties of the empty image object.
- For more information about defining the properties of an empty image object, see *Importing External Content* in the Exstream Design and Production documentation.
6. From the Menu bar, select **File > Save**.

3.9.2 Converting Inline Image Placeholder Variables to Empty Image Objects

If you have inline image placeholder variables within your old and current design pages, you can convert them to empty image objects for use in a Live document.

Caution: If you have existing design pages that you are preparing for a Live document, you must convert all inline image placeholder variables (placeholder variables not referenced by an empty image object) to empty image objects. Exstream Live does not support inline image placeholder variables.

An advantage of converting inline image placeholder variables to empty image objects is the ability to see design pages that closely resemble the expected output. For example, converting inline placeholder variables to empty image objects lets you see how the text will shift around

the image so that you can determine whether you want to make any changes to the image size or to the surrounding text.

To convert inline placeholder variables to empty image objects:

1. In Designer, right-click the inline placeholder variable and select **Convert to Empty Image Object**.

A message box appears asking if you want to convert all inline image placeholders.

2. Select one of the following options:

- **Yes**—This converts all supported inline image placeholders to empty image objects.
- **No**—This converts only the selected inline image placeholder.

The message box closes and the results appear on the design page.

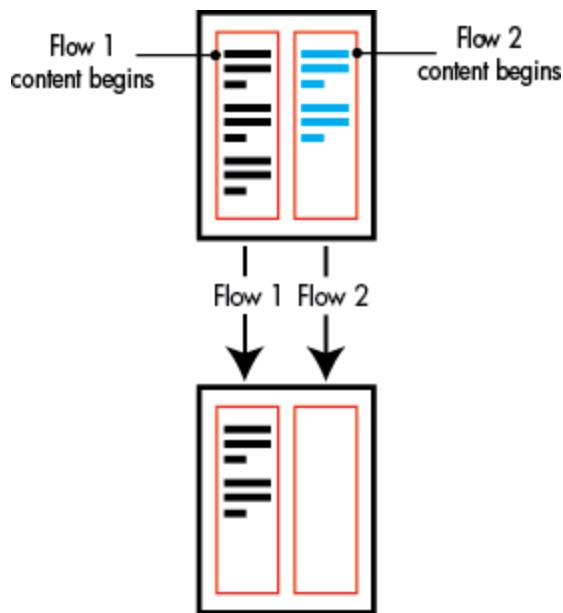
3.10 Using Flow Frames in Live Documents

Just as you can in traditional Exstream Design and Production designs, you can use flow frames in Live documents to contain overflow content from tables, text boxes, section objects, and paragraph objects. However, keep in mind the following design considerations:

- When you include an editable area that can flow in a Live document, the total number of pages in the Live document can change when an end user edits the content.
- When you include an editable area that can flow in a Live document, and you use DLF output produced by that Live document in a fulfillment application, the total number of campaign messages in the final fulfillment might change if an end user has edited the content in the Live document. The number of campaign messages that are included in the final fulfillment is determined by the amount of available space in a flow frame.
- When you use flow targets to designate content to flow to a specific frame in a Live document, and you use DLF output produced by that Live document in a fulfillment application, you must make sure that the fulfillment application also has flow frame targeting enabled.

- If a Live design includes flow targets, multiple flow frames on the same page, and content that overflows to a flow page, then the engine adds content to multiple targeted flow frames on the first page. For example, suppose that you have one targeted flow frame that includes five paragraphs and a second targeted flow frame that includes a single paragraph. Even if the content in the first targeted flow frame spans across multiple pages, the engine will add content in both flow frames beginning on the first page.

Example of a design with multiple targeted flow frames



For more information about designating content to flow to a specific frame, see *Designing Customer Communications* in the Exstream Design and Production documentation.

For information about how marketing messages use flow frames, see *Managing Marketing Messages* in the Exstream Design and Production documentation.

3.11 Using Tables of Contents and Indexes in Live Documents

Tables of contents and indexes are document components that can help improve the usability of your Live documents. For example, if the Live document is long, a table of contents can help orient both end users and customers as they read it. Also, end users can press and hold CTRL and click entries in the table of contents to navigate the document.

Keep in mind that the page number value displayed in a table of contents or index entry corresponds to the value of the 'SYS_PagePrintedValue' variable. Therefore, if you want to

include a table of contents or index in a Live document, you must use the 'SYS_PagePrintedValue' variable for page numbering.

For more information about 'SYS_PagePrintedValue' and other page numbering variables, see ["Unique Behaviors of Page Numbering System Variables in Live Documents" on page 429](#).

Keep in mind the following behaviors if you include tables of contents or indexes in your Live designs:

- Although tables of contents and indexes that you include in a Live document will be updated in LiveEditor based on the selections an end user makes, the end user cannot edit the content of these objects (as you can with other Designer objects) in LiveEditor. For example, you cannot manually change the value of a table of contents entry in LiveEditor.
- End users cannot edit or add markers for tables of contents or indexes. Therefore, if end users will make selections that cause additional content to be included in a customer's Live document, you must add table of contents or index markers for this content when you design the Live document.
- If you are using paragraph objects that are listed in a table of contents and you allow end users to make free-form changes to the text in the paragraphs, keep in mind that LiveEditor does not delete the paragraph entry from the table of contents when an end user deletes a paragraph. If you want to allow the end user to remove paragraphs and have LiveEditor update the table of contents automatically, you can set up a content selection group that includes the paragraphs.

For more information about using section and paragraph objects, see *Designing Customer Communications* in the Exstream Design and Production documentation.

For more information about allowing end users to make free-form changes to text, see ["Allowing End Users to Make Free-Form Changes to Text" on page 39](#).

For more information about using a content selection group, see ["Using a Content Selection Group to Dynamically Select Content" on page 100](#).

- If you use the Live document in a fulfillment process that adds documents, new table of contents and index entries are not added. However, the page numbers in the table of contents and index will be updated to reflect any changes that result from adding documents.
- When you select **Hyperlink** in the properties of a table of contents placeholder included in a Live document, navigation from the table of contents in LiveEditor is also enabled, and an end user can press and hold CTRL and click a heading in the table of contents to go directly to the relevant section of the Live document.

For more information about allowing end users to navigate using the table of contents, see ["Allowing End Users to Navigate From the Table of Contents" on page 241](#).

- During fulfillment, if documents are inserted before a Live document containing a table of contents, the table of contents is not removed from the Live document automatically. You must set up rules to exclude the pages containing the table of contents. Likewise, if documents are inserted after a Live document containing an index, you must set up rules to exclude the index pages if you do not want them to appear in the middle of the new document.

For more information about tables of contents and indexes, see *Designing Customer Communications* in the Exstream Design and Production documentation.

3.12 Designing Live Documents for Multiple Languages

If you are designing a Live document that will be sent to end users who speak different languages, or if end users want to send the final output to customers who speak different languages, you can use language layers to reuse the same document and page design, while still providing information to end users and customers in different languages as necessary. Language layers are design objects that let you create multiple layers of localized content on top of a base design.

For example, suppose you are setting up a Live document that is being sent to end users in multiple countries. Instead of having to create a separate DLF file for each language, you can create a single Live document that allows each end user to choose his or her language. In this scenario, you can use a simple device that allows end users to choose their language—such as a series of country flags at the top of the page. Each flag can be an interactive area that changes the value of the 'SYS_LanguageCustomer' variable so that choosing an American flag displays English content, choosing a French flag displays French content, and so on.

Moreover, you can use recipient processing to ensure that customers receive the final output from a Live document in the proper language. For example, suppose you are designing an insurance form that end users will send to policyholders. However, regulations require that policyholders in certain states must also receive a Spanish-language version of the form. Using recipient processing, you can trigger a Spanish-language duplicate copy for policyholders in the affected states.

For more information about using recipient copies, see [“Allowing End Users to Add and Modify Copies of Documents” on page 178](#).

For more information about setting up language layers in Design Manager, see *Designing Customer Communications* in the Exstream Design and Production documentation.

This section discusses the following topics:

- [“Adding Language Layers to a Live Document” on the next page](#)
- [“Creating Live Documents for Complex Text Languages” on the next page](#)
- [“Creating DLF Output for Live Documents That Contain Multiple Language Layers” on page 70](#)
- [“Adding Content to Language Layers in a Live Document” on page 71](#)
- [“Using Language Layers in a Live Document: The End User Experience” on page 72](#)

3.12.1 Adding Language Layers to a Live Document

Before you can add a language layer to a page, you must create the language object for that language. Although you add the content for each language in Designer, each language must first have a corresponding language layer in Design Manager.

Language objects are defined in the **System Settings**, so in order to use multiple languages in a Live document, you must enable multiple languages for your system.

For more information about defining languages, see *System Administration* in the Exstream Design and Production documentation.

To add a language layer to a page:

1. In Design Manager, from the Library, drag the page to the Property Panel.

The page opens in the Property Panel.

2. Click the **Languages** tab.

3. In the **Languages** area, click .

The **Select language** dialog box opens.

4. From the list of languages, select the appropriate language.

5. Click **OK**.

The **Select language** dialog box closes.

6. If the customer language is not defined in the 'SYS_LanguageCustomer' system variable, and you want to send the default language, select the **Send default language if customer language does not exist** check box.

For information about defining a customer language in the 'SYS_LanguageCustomer' system variable, see *Designing Customer Communications* in the Exstream Design and Production documentation.

7. From the Menu bar, select **File > Save**.

8. Drag the page to the Edit Panel.

The page opens in Designer for you to add content.

3.12.2 Creating Live Documents for Complex Text Languages

The design considerations for Live documents that use complex text languages (such as Arabic, Farsi, and Hebrew) are the same as those for any other Exstream application using complex text languages—that is, you must update your **System Settings**, specify font settings on the

application object and on the output object, and configure settings on the language object. For a complete description of the steps you must follow when setting up an application to use complex text languages, see *System Administration* in the Exstream Design and Production documentation.

To include complex text languages in a Live document, however, you must configure some additional settings. The following table provides a high-level overview of the steps (along with links to guides and sections that discuss them) that you must follow when using complex text languages in your Live designs:

High-level overview for using complex text languages in Live applications

Step	Explanation	Links
1. In the documentation, follow the instructions for configuring language settings for complex text languages.	If you are using complex text with Live, you need to start with the "Configuring Language Settings for Complex Text Languages" chapter in <i>Exstream System Administration</i> , which guides you through the process of setting up applications to use complex text languages.	<i>System Administration</i> in the Exstream Design and Production documentation
2. On the Live Settings object, set the packaging options to reference all fonts.	With complex text layout enabled in your database, you should always reference—rather than embed—fonts in Live documents because embedded fonts will cause a significant increase in the size of the DLF file. (Note that this limitation applies only to DLF files; you can safely embed complex text fonts for all other outputs.)	"Controlling How Fonts are Handled When Packaging a DLF File" on page 251
3. In the fulfillment application, make sure that all of your font settings match those in your design application.	As with any Live fulfillment application, you must ensure that you include all resources (especially those for fonts) that will be necessary to properly process your Live document during the fulfillment engine run.	"Defining Application Resources for Live Document Processing" on page 379
4. In the documentation, review limitations and notes about special use cases.	Before you implement complex text layout in your database, review the "Design Differences" section so that you can avoid any pitfalls.	<i>System Administration</i> in the Exstream Design and Production documentation

3.12.3 Creating DLF Output for Live Documents That Contain Multiple Language Layers

After you have enabled language layer use in a Live document, you must complete the following steps to specify the time at which language layers will be used in your Live output:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Language layer use** drop-down list, select one of the following options:

- **Initial engine run only**—Select this option to use language layers to determine content in the Live document, but not to allow end users to select between language layers in LiveEditor.
 - **Initial Engine and Interactive Editor**—Select this option to use language layers to determine content in the Live document, and to allow end users to select between language layers in LiveEditor.
 - **Initial Engine, Interactive Editor, and Final Engine**—Select this option to use language layers to determine content in the Live document, to allow end users to select between language layers in LiveEditor, and to allow language layers to be used during fulfillment.
4. From the Menu bar, select **File > Save**.

For more information about setting up language layers in Design Manager, see *Designing Customer Communications* in the Exstream Design and Production documentation.

3.12.4 Adding Content to Language Layers in a Live Document

After you have created a language layer and specified how it will be processed for your Live document, you can access the language layer and add content in Designer.

To add content for a particular language layer:

1. In Designer, select the object to which you want to add language-specific content.

2. On the Standard toolbar, click .

The **Select the language to edit** dialog box opens.

3. Clear the **Show existing languages only** check box.

If you select the **Show existing languages only** check box, you limit the layers that you can select to only those with objects on them. If you clear the check box, all of the available language layers appear.

4. From the **Select the language to edit** drop-down list, select the language layer to which you want to add language-specific content.

5. Click **OK**.

The **Select the language to edit** dialog box closes and the language layer you selected opens in Designer.

6. Use the Designer tools to add content to the language layer.

7. From the Menu bar, select **Edit > Save**.

For more information about adding language layers in Designer, see *Designing Customer Communications* in the Exstream Design and Production documentation.

3.12.5 Using Language Layers in a Live Document: The End User Experience

Depending on how you set up your Live document, end users might have a number of ways to access different language layers in LiveEditor. Even if you use the 'SYS_LanguageCustomer' variable to automate the process of choosing languages in LiveEditor, end users can use the following basic process to switch between languages:

1. An end user opens a Live document that contains language layers.
2. From the LiveEditor Menu bar, the end user selects **View > Language Layer**.
3. From the **Select the language to edit** dialog box, the end user selects an available language.

The language layer in LiveEditor switches to the language that the end user selected.

3.13 Defining Layouts for Electronic Devices That Can Be Delivered from Live Documents

Each page in Exstream allows for two types of design: A standard design, which uses fixed-dimension layouts that are well suited for print-based layouts, such as PDF; and container designs, which use relative layouts that can automatically adjust and flow to accommodate a variety of electronic devices (for example, smartphones, laptops, and tablets). You can use container designs in a Live document to allow end users to update designs so that they can deliver HTML-based output, including HTML, HTML (email), and Multi-Channel XML (with the HTML format). While you can include a standard design and also container designs in your Live document, end users can see and interact with only the objects that are within the design area in your standard design.

As you create Live documents that include a standard design and also container designs, keep in mind the following:

- Fulfillment applications must include any application resources that are used by the standard design and any container designs. However, if you want to allow end users to deliver container-based output, your fulfillment application must also include an output object that is set up to deliver container designs and any associated container design labels.

For more information about setting up a fulfillment application, see “[Processing Live Documents](#)” on page 360.

For more information about setting up HTML, HTML (email), or Multi-Channel XML output to support container designs, see *Creating Output* in the Exstream Design and Production documentation.

- Standard designs and container designs share the same objects. Because these objects are shared, changes that users make to the objects from the standard design—including text, images, variables, tables, and charts—will be made to the same object in the container design.
- If an end user resizes an object (such as a logo) in a Live document, the object will also resize in the container design.
- Although objects are shared between designs, some object properties are specific to a design. For example, objects are positioned differently between the standard design and any container designs. If an end user changes the position of an object (such as a logo or a picture), that change will be reflected only in the standard design and it does not affect the placement of the object in the container design.

For more information about which object properties are shared across the standard design and container designs, see *Designing Customer Communications* in the Exstream Design and Production documentation.

- If you must change the layout of a container design, you must recreate both the Live document and the fulfillment application in order to reflect the layout changes in the final output. End users cannot change the layout of a container design from the Live document.
- If you use an object in a container design but you do not use the object in the design area of the standard design, end users will not see that object in the Live document. For example, if you use a text box in a container design, but the text box is on the pasteboard of the standard design, end users will not be able to see or edit that text box in the Live document.

The following table discusses important considerations for using shapes, images, and dynamically imported content in a Live document that will be fulfilled to both page-based output and container-based output:

Live document feature	Container design considerations
Shapes, charts, images, and image selectors	<ul style="list-style-type: none">• It is a best practice to avoid placing editable objects in absolute position containers. Since end users cannot view the container design as they make changes, they cannot see how their changes impact the appearance or flow of objects in a container design. Objects in absolute position containers can grow to overlap other objects in the design or push adjacent objects into different positions (depending on your container designs). Keep in mind that if end users can edit objects that are stored in an absolute position container, you might receive unexpected results in the final output.• Since container sizes are preset during the initial engine run, it is a best practice to create enough space in the original design to accommodate any changes to non-flowing items (such as images or shapes). Creating this space in the original design ensures that these objects appear as expected in the final output. For example, if you set up a Live document to allow end users to select an image in the design, you should create the initial empty image or image selector object and use the largest potential width in order to provide enough space in the container during the initial engine run. This would therefore accommodate any changes that the end user could make during a LiveEditor session.

Live document feature	Container design considerations
Dynamically importing content	<ul style="list-style-type: none">Container designs support dynamic content import using only inline placeholder variables or placeholder images.Container designs do not support placeholder documents. If placeholder documents are attached to the content in LiveEditor, these documents are delivered only with the print-based outputs (with the standard design) and will not be included when the content is produced as HTML-based output (with the container designs).

3.13.1 Editing and Producing Output from a Live Document That Includes Container Designs: The End User Experience

This section describes what end users will see when they edit Live documents that include container designs.

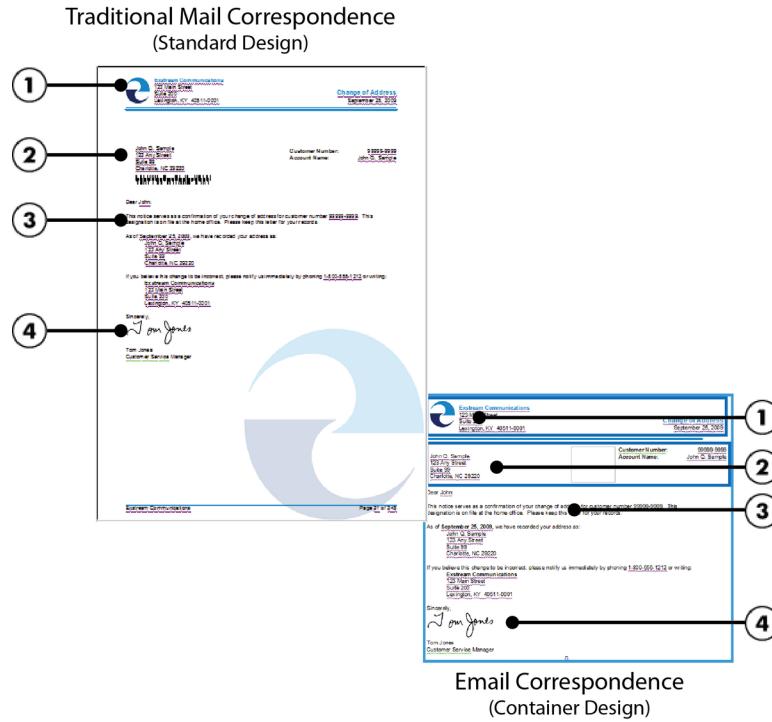
Suppose that you are designing a correspondence document for an insurance company. You want agents in the field to update customer data and personalize their interactions with their customers so that customers will receive an email instantly, but also receive a copy of the interaction in the mail along with any brochures or new customer information that they might require.

To design this kind of correspondence document, you must set up the Live document to include a traditional mail correspondence design (standard design) that not only includes the customer correspondence, but also includes placeholder documents—such as product brochures and new customer documents (if applicable). You must also set up the Live document to include an email correspondence design (container design) by using many of the same objects that you included in the mail correspondence design. This email correspondence design can then be delivered as an initial email confirmation to the same customers who received the traditional mail correspondence.

When the end user receives the Live document, he or she will be able to open and view only the content that is included in the traditional mail correspondence design. However, since the objects in the mail correspondence design are shared with the email correspondence design, the changes that end users make to customer information and content will update the same content in both designs.

The following graphic illustrates how the changes that the end user makes while viewing and editing the Live document affect the hidden container design, and ultimately the final output.

Example of how updates to a Live document affect the hidden container design



Label	Interactions with the Live document	Effects on the container design
1	The end user resizes the logo.	When the end user resizes the logo in the Live document, the logo also resizes in the container design. Additionally, the resize also affects the placement of the adjacent address block in the container design layout for email output.
2	The end user updates the customer address information.	When the end user updates the customer address data, the data also updates across the container design for email output.
3	The end user enters a personalized response and adds variables to provide customer-specific data in the response (such as name, address, and membership status).	When the end user updates the response, the text and variable updates are also applied to the container design for email output.
4	The end user uses an image selector to select his or her signature and/or uploads a custom signature graphic.	When the end user selects or uploads a signature graphic, the appropriate signature graphic is also applied to the container design for email output.

To finalize all of the Live document changes and to deliver content, the end user clicks the button that submits the Live document to the fulfillment application that is set up to deliver standard designs in PDF output and to deliver the container design to HTML (email) output. The customer will then receive an email copy of the correspondence and separately receive a mail copy of the same correspondence accompanied by any applicable placeholder documents.

Chapter 4: Controlling End User Editing Using Selection Controls

One way to provide a controlled, intelligent editing experience for end users is to leverage the selection control objects available with Exstream Live. Selection controls are objects such as drop-down lists, radio buttons, check boxes, and calendars that are often used in documents to collect data. Exstream Live also provides some types of selection control objects that are unique to Live documents and are designed to support customizations specific to Live documents. For example, you can add content selection groups that allow end users to select text or images to include in a specific place in a Live document. Other Live-specific selection controls you can add are custom buttons used to launch specific actions.

Example of a page that uses selection controls to control editing

New enrollee information

BASIC INFORMATION

Promotion/Discount code?

First name: MR:
Last name: Suffix:
Address: Add near:
City: State:
Zip code: Email:

WEBSITE ACCOUNT SETUP

Default login: Login security image:

PREFERENCES

Preferred method of contact:

Email
 Cell phone
 Home phone
 Text message

Preferred branch:

Louisville
 Lexington
 Frankfort
 Bowling Green

Preferred payment due date:

This customer requires a follow-up call from a customer service representative. Follow-up is needed on the enrollment process or other specific questions.

This customer requires a follow-up call from an accounting representative. Follow-up is needed on billing and payment plan options.

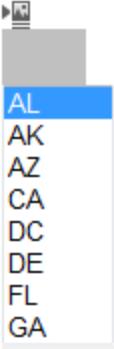
This customer requires a follow-up call from a local branch representative. Follow-up is needed on specific branch service questions.

Using selection controls to control end user editing allows you to add another level of intelligence to a Live document. Because most of the selection controls available in Live documents allow you to record a specific value when end users make a selection, you can use that data to drive other processes. For example, if an end user selects the "Male" radio button on an enrollment form, you can use the recorded value to fire a function or set other values in the Live document automatically. Selection controls are often used on interview pages to collect data that drives the assembly and customization of a document.

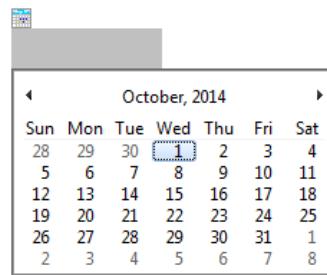
For more information about using interview pages, see “[Using Interview Functionality to Drive Data Changes](#)” on page 123.

The number and types of selection controls you use in a Live document depend on the type of document you are creating and how you want end users to interact with it. For example, in some types of Live documents, such as a letter or a proposal, you might choose to include very few, if any, selection controls. In other types of Live documents, such as enrollment documents, you might choose to use selection controls exclusively, allowing end users to make selections that will drive other processes. The following table lists the types of selection controls available to you as you design a Live document, and explains when you might choose to use each type.

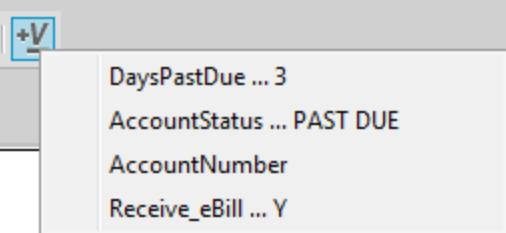
Selection control objects in Live documents

Selection control	Description	Example
Check boxes	Like radio buttons, check boxes allow end users to make selections from a set of options. Use check boxes when end users can select one or more of the options.	<p>Preferred method of contact</p> <p><input type="checkbox"/> Email <input type="checkbox"/> Cell phone <input type="checkbox"/> Home phone <input type="checkbox"/> Text message</p>
Radio buttons	Like check boxes, radio buttons allow end users to make a selection from a set of options. Use radio buttons when end users can select only one option from a group of choices.	<p>Preferred branch</p> <p><input type="radio"/> Louisville <input type="radio"/> Lexington <input type="radio"/> Frankfort <input type="radio"/> Bowling Green</p>
Drop-down lists	Drop-down lists provide a controlled way for end users to customize text. You can use drop-down lists to allow end users to replace a specified area of content with pre-set text. The options in a drop-down list can be either static or variable.	<p>State:</p> 

Selection control objects in Live documents, continued

Selection control	Description	Example																																																	
Image selectors	Image selectors are objects that allow end users to customize the images that appear in a Live document. Unlike allowing end users to upload an image of their choosing, image selectors allow you to provide a set of images from which end users can select an image to include.	 <p>Login security image:</p>																																																	
Content selection groups	Content selection group objects allow you to add automation to a Live document by enabling end users to select sections of content that should be included from a list of possible content, while automatically hiding the content that is not selected. Content selection groups work particularly well when you want end users to select one or more pre-written paragraphs that should be included at a given point in a Live document.	<p><input type="checkbox"/> This customer requires a follow-up call from a customer service representative during the enrollment process or other specific questions.</p> <p>Pick 1 to 3 (0 picked)</p> <p><input type="checkbox"/> Customer service follow up? <input type="checkbox"/> Local representative follow-up? <input type="checkbox"/> Accounting follow-up?</p> <p>OK Cancel</p>																																																	
Buttons	Buttons are objects that launch actions when an end user clicks them. You can create buttons with custom text and custom actions to add intelligence and automation to a Live document.	<p>Submit for approval</p>																																																	
Calendars	Calendars are graphical representations of dates to which end users can browse and select a specific date.	<p>Payment due date:</p> <p></p> <table border="1"> <caption>October, 2014</caption> <thead> <tr> <th>Sun</th><th>Mon</th><th>Tue</th><th>Wed</th><th>Thu</th><th>Fri</th><th>Sat</th></tr> </thead> <tbody> <tr> <td>28</td><td>29</td><td>30</td><td>1</td><td>2</td><td>3</td><td>4</td></tr> <tr> <td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td></tr> <tr> <td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td></tr> <tr> <td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td></tr> <tr> <td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td>1</td></tr> <tr> <td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td></tr> </tbody> </table>	Sun	Mon	Tue	Wed	Thu	Fri	Sat	28	29	30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4	5	6	7	8
Sun	Mon	Tue	Wed	Thu	Fri	Sat																																													
28	29	30	1	2	3	4																																													
5	6	7	8	9	10	11																																													
12	13	14	15	16	17	18																																													
19	20	21	22	23	24	25																																													
26	27	28	29	30	31	1																																													
2	3	4	5	6	7	8																																													

Selection control objects in Live documents, continued

Selection control	Description	Example
Form fields	Form fields are objects that allow you to control the type and length of content that is entered in an editable area or in an editable variable.	
Variable Palette	You can provide a customized Variable Palette that end users can interact with to select and insert variables into a Live document. Like the Variable Palette in Design Manager or Designer, the Variable Palette in LiveEditor allows end users to browse and select variables. However, you can limit the variables that appear in the Variable Palette to make it easier for end users to find and select the appropriate variable.	

This chapter discusses the following topics:

- “Using Check Boxes and Radio Buttons to Illustrate Choices” on the next page
- “Using Drop-Down Lists to Show Options” on page 88
- “Using an Image Selector to Allow End Users to Choose From a Set of Images” on page 91
- “Using a Content Selection Group to Dynamically Select Content” on page 100
- “Using Buttons to Launch Actions” on page 106
- “Using a Calendar to Allow End Users to Choose Dates” on page 111
- “Using Form Fields to Control Data Entry” on page 113
- “Setting Up a Variable Palette to Allow End Users to Insert Variables” on page 119
- “Controlling the Visibility of Activation Buttons in LiveEditor” on page 120
- “Changing the Fonts Used in LiveEditor Controls” on page 121

4.1 Using Check Boxes and Radio Buttons to Illustrate Choices

Check boxes and radio buttons are common selection controls that allow end users to see a list of options and choose one by selecting the check box or radio button. Check boxes should be used to control options in which end users can make one or more selections, and radio buttons should be used to control options in which end users can make only one selection.

Example of check boxes

Preferred method of contact

- Email
- Cell phone
- Home phone
- Text message

Example of radio buttons

Preferred branch

- Louisville
- Lexington
- Frankfort
- Bowling Green

4.1.1 Setting Up a Check Box

To set up a check box, you must complete the following tasks:

1. [“Adding a Check Box to a Page” on the next page](#)
2. [“Defining the Variable Settings of the Check Box” on the next page](#)

You can also complete the following optional task as needed:

- [“Customizing the Appearance of the Check Box” on page 82](#)

Adding a Check Box to a Page

In Designer, on the Drawing Objects toolbar, click . A check box is placed on the page. You can use the tools in Designer to move and resize the check box as needed.

Defining the Variable Settings of the Check Box

You use the options in the variable settings area of the check box properties to specify the variable that will be populated when the check box is selected, and how the value of that variable is stored.

Before beginning this task, you must have created a variable designed to store the value of a check box when it is selected. If you want to allow end users to select more than one check box, you must create an array variable.

For more information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To define the variable settings of a check box:

1. In Designer, select the check box and click .

The **Checkbox Properties** dialog box opens.

2. Click the **Button** tab.

3. Use the **Variable** box to specify the variable that will be populated with a value when an end users selects the check box. The variable should be a Boolean, string, or integer variable.
 - a. Click .

The **Select Variable** dialog box opens.

- b. Select the variable and click **OK**.

The **Select Variable** dialog box closes and the variable you selected appears in the **Variable** box.

4. If you specified an array in the **Variable** box, enter the index value of the array element that will be populated when the check box is selected in the **Array element** box. Enter 0 if you want the element to be populated automatically.

5. Use either the **Value if clicked** box or the **Array containing selection values** box to specify the value stored when the check box is selected.

To	Do this
Store a static specified value	In the Value if clicked box, enter the value you want to populate the specified variable when the check box is selected. This box is active only if you specified a string or integer variable.
Store a variable-driven value	a. In the Array containing selection values box, click  . The Select Variable dialog box opens. b. Select the variable that provides the values and click OK . The Select Variable dialog box closes and the variable you selected appears in the Array containing selection values box.

6. From the Menu bar, select **Edit > Save**.

If you use an array variable to store the value of the check box, and the check box is placed in an object that can repeat dynamically (such as a table row), the elements in the array are populated with values in the order that the check boxes appear. For example, if the check box is located in a repeating table row, the first element in the array contains the first check box value, the second element contains the second, and so on.

If you use array variables to set up the check box values, and the check box is placed in an automated data-section driven table, all of the variables used to define the check box must be associated with the data section. Therefore, if you cannot map the variable that stores the check box value in the data section, you must use the VARSCOPE switch in a control file to specify the section that is associated with the variable.

For more information about the VARSCOPE switch, see *Switch Reference* in the Exstream Design and Production documentation.

Customizing the Appearance of the Check Box

1. In Designer, select the check box and click .
- The **CheckBox Properties** dialog box opens.
2. Click the **Button** tab.
3. From the **Button style** drop-down list, select one of the options to specify the appearance of the check box:

Button style options

Option	Sample
Drawn (2d check)	
Drawn (3d check)	
Drawn (X check)	
Drawn (wide X check)	
Drawn (filled check)	
Standard images	
Images	Varies

The graphical representation of the check box in Designer does not change based on your selection from the **Button style** drop-down list; the differences are visible only in LiveEditor.

4. If you selected **Images** from the **Button style** drop-down list, additional options appear at the bottom of the dialog box. Use the options to specify images that are used for each state of the check box (up, down, and hover).
 - a. Double-click the **Up** box.

The **Import an Image** dialog box opens.

 - b. Browse to and select the image that you want to be used for the check box when it is not selected.
 - c. Click **Open**.

The **Import an Image** dialog box closes and the image you selected appears in the **Up** box.

 - d. Repeat step a through step c for the **Down** box (image when the check box is selected) and **Hover** box (image when an end user hovers the pointer over the check box).
5. From the **Box style** drop-down list, select one of the options to specify the appearance of the border around the check box.

Box style options

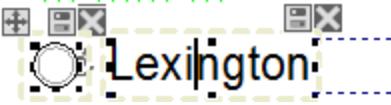
Option	Sample
Flat	<input type="checkbox"/>
3d	<input checked="" type="checkbox"/>
3d thin	<input type="checkbox"/>

6. Click **OK**.

The **CheckBox Properties** dialog box closes. If you specified images to be used in place of the standard check box options, or if you customized the border of the check box, the placeholder check box reflects your selections.

7. After you have defined the basic check box properties, you can use some of the other properties in Designer to customize its appearance. The following table describes some of the common tasks you might want to complete to further customize the check box:

Common check box formatting tasks

If you want to	Do this
Change the color and style of the check or the check box background	Use the options on the Lines and Fill tab of the CheckBox Properties dialog box. For more information about using the options on the Lines and Fill tab, see <i>Designing Customer Communications</i> in the Exstream Design and Production documentation.
Adjust the check box behavior so that end users can select text adjacent to the check box in order to select the check box	Drag the dashed line surrounding the check box itself so that it encompasses the text. Check box and text in design 

8. From the Menu bar, select **Edit > Save**.

4.1.2 Setting Up a Radio Button

To set up a radio button, you must complete the following tasks:

1. [“Adding a Radio Button to a Page” below](#)
2. [“Defining the Variable Settings of the Radio Button” below](#)

You can also complete the following optional task as needed:

- [“Customizing the Appearance of the Radio Button” on the next page](#)

Adding a Radio Button to a Page

In Designer, on the Drawing Objects toolbar, click  . A radio button is placed on the page. You can use the tools in Designer to move and resize the radio button as needed.

Defining the Variable Settings of the Radio Button

You use the options in the variable settings area of the radio button properties to specify the variable to be populated with a value when the radio button is selected, and how the value of that variable is stored.

Before beginning this task, you must have created a variable designed to store the value of the radio button when it is selected.

For more information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To define the variable settings of a radio button:

1. In Designer, select the radio button and click .
2. Click the **Button** tab.
3. Use the **Variable** box to specify the variable that will be populated with a value when an end users selects the radio button. The variable should be a string or integer variable. Do not use Boolean variables for radio buttons. Since Boolean variables are either true or false, all radio buttons would have the same value.

- a. Click .

The **Select Variable** dialog box opens.

- b. Select the variable and click **OK**.

The **Select Variable** dialog box closes and the variable you selected appears in the **Variable** box.

4. If you specified an array in the **Variable** box, enter the index value of the array element that will be populated when the radio button is selected in the **Array element** box. Enter 0 if you want the element to be populated automatically.
5. Use either the **Value if clicked** box or the **Array containing selection values** box to specify the value stored when the radio button is selected.

To	Do this
Store a static specified value	In the Value if clicked box, enter the value you want to populate the specified variable when the radio button is selected. This box is active only if you specified a string or integer variable.
Store a variable-driven value	<ol style="list-style-type: none">a. In the Array containing selection values box, click  . The Select Variable dialog box opens.b. Select the variable that provides the values and click OK. <p>The Select Variable dialog box closes and the variable you selected appears in the Array containing selection values box.</p>

6. From the Menu bar, select **Edit > Save**.

If you use an array variable to store the value of the radio button, and the radio button is placed in an object that can repeat dynamically (such as a table row), the elements in the array are populated with values in the order that the radio buttons appear. For example, if the radio button is located in a repeating table row, the first element in the array contains the value for the radio button in the first row, the second element contains the value for the button in the second row, and so on.

If you use array variables to set up the radio button values, and the radio button is placed in an automated data-section driven table, all of the variables used to define the radio button must be associated with the data section. Therefore, if you cannot map the variable that stores the radio button value in the data section, you must use the VARSCOPE switch in a control file to specify the section that is associated with the variable.

For more information about the VARSCOPE switch, see *Switch Reference* in the Exstream Design and Production documentation.

Customizing the Appearance of the Radio Button

1. In Designer, select the radio button and click .

The **RadioButton Properties** dialog box opens.

2. Click the **Button** tab.
3. Use the **Button style** and **Box style** options to specify the appearance of the radio button:

To	Example	Do this
Use a three-dimensional radio button		<ul style="list-style-type: none"> a. From the Button style drop-down list, select Drawn (radio). b. From the Box style drop-down list, select one of the options to specify the appearance of the border around the radio button.
Use a two-dimensional radio button created by Designer		From the Button style drop-down list, select Standard images .
Use an image you select	Varies	<ul style="list-style-type: none"> a. From the Button style drop-down list, select Image. Additional options appear at the bottom of the dialog box. b. Double-click the Up box. The Import an Image dialog box opens. c. Browse to and select the image that you want to be used for the radio button when it is not selected. d. Click Open. The Import an Image dialog box closes and the image you selected appears in the Up box. e. Repeat step b through step d for the Down box (image when the radio button is selected) and Hover box (image when an end user hovers the pointer over the radio button).

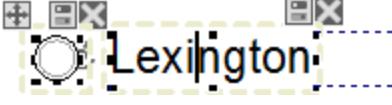
The graphical representation of the radio button in Designer does not change based on your selection from the **Button style** drop-down list; the differences are visible only in LiveEditor.

4. Click **OK**.

The **RadioButton Properties** dialog box closes. If you specified images to be used in place of the standard radio button options, the radio button on the page reflects your selections.

5. After you have defined the basic radio button properties, you can use some of the other properties in Designer to customize its appearance. The following table describes some of the common tasks you might want to complete to further customize the radio button:

Common radio button formatting tasks

If you want to	Do this
Change the background of the radio button	Use the options on the Lines and Fill tab of the RadioButton Properties dialog box. For more information about using the options on the Lines and Fill tab, see <i>Designing Customer Communications</i> in the Exstream Design and Production documentation.
Adjust the radio button behavior so that end users can select text adjacent to the radio button in order to select the radio button	Drag the dashed line surrounding the radio button itself so that it encompasses the text. Radio button and text in design 

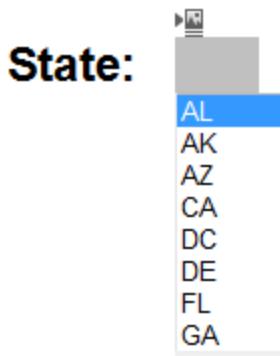
- From the Menu bar, select **Edit > Save**.

4.2 Using Drop-Down Lists to Show Options

Drop-down lists allow you to provide a pre-set list of options from which end users can select text for inclusion in the Live document. Drop-down lists provide a more controlled alternative than allowing end users to make free-form changes to the text in a document: with a drop-down list, end users can customize the text using only approved substitutions.

For more information about the ways you can allow end users to change text, see “[Allowing End Users to Change Text](#)” on page 39.

Example of a drop-down list



The options that appear in a drop-down list can be either static or variable text. If you want to include variable items on the drop-down list, you must set up an array variable that provides the list items before beginning this task.

For more information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To use a drop-down list to show options:

1. In Designer, do one of the following to select the text that you want to allow end users to change:

To	Do this
Allow end users to change of all the text within a text box	<ol style="list-style-type: none">Select the text box.Click  . <p>The Text Properties dialog box opens.</p>
Allow end users to change only a specific paragraph of text in a text box or table cell	<ol style="list-style-type: none">Select the lines or paragraphs of text.Right-click the selected text and select Text paragraph properties. <p>The Text paragraph properties dialog box opens.</p>

2. Click the **Interactive** tab.
3. From the **Content change** drop down list, select either **Change is optional** or **Change is required**, as needed.

Additional options become available.
4. Select the **Text edit** check box.

Additional options become available.
5. From the **Use control to select content** drop-down list, select either **Static list** or **Variable list**.

Additional options become available.
6. Use the additional options to specify the contents of the drop-down list.

To	Do this
Populate the drop-down list with static options	<ol style="list-style-type: none">Click . The List Items dialog box opens.Click .In the Text of selected item box, enter the first entry for the drop-down list.Repeat step b through step c to add as many items as needed to the list. <p>You can use the  and  buttons to promote or demote items in the list. If you want to sort the list items alphabetically, you can select the Sort list items check box on the object properties dialog box to allow the items to be sorted automatically.</p> <ol style="list-style-type: none">Click OK. The List Items dialog box closes.
Populate the drop-down list with dynamic options	<ol style="list-style-type: none">In the Display values for list population box, click . The Select Variable dialog box opens.Select the variable that provides the content of the drop-down list and click OK. <p>The Select Variable dialog box closes and the variable you selected appears in the Display values for list population box.</p>

7. Use the **Selection index variable** box to specify the variable that stores the index of the drop-down list item that the end user selects. This variable must be an array variable if you want to allow end users to make multiple selections from the list.

a. Click .

The **Select Variable** dialog box opens.

b. Select the variable that stores the value and click **OK**.

The **Select Variable** dialog box closes and the variable you selected appears in the **Selection index variable** box.

8. If you want the list items to be sorted automatically so that they appear in alphabetical order, select the **Sort list items** check box.

9. If you want to allow end users to select multiple items on the drop-down list, select the **Allow multiple select** check box.

10. Click **OK**.

The object properties dialog box closes.

11. After you have defined the basic drop-down list properties, you can use some of the other properties in Designer to customize its appearance. The following table describes some of

the common tasks you might want to complete to further customize the drop-down list:

Common drop-down list formatting tasks

If you want to	Do this
Specify the font and font size used in a static drop-down list	Use the Fonts to use in LiveEditor controls option on the setting object. For more information about customizing the fonts used in LiveEditor controls, see "Changing the Fonts Used in LiveEditor Controls" on page 121 .
Control whether activation buttons appear on drop-down lists to help end users see them	Use the options on the Editor Framework tab of the setting object to control the visibility of activation buttons. For more information about controlling whether activation buttons appear for drop-down lists, see the "Controlling the Visibility of Activation Buttons in LiveEditor" on page 120 .

12. From the Menu bar, select **Edit > Save**.

4.3 Using an Image Selector to Allow End Users to Choose From a Set of Images

An image selector is a selection tool that displays all the possible images an end user can insert at a specific location in a DLF. You can use two methods to set up image selectors to control the images end users can add to DLFs:

- ["Including Images as Part of the DLF File" on the next page](#)
- ["Dynamically Importing Images From a Shared Location" on page 95](#)

Both methods ensure that end users do not have to maintain images on their own computers. To select an image, end users can single-click the image selector box to activate arrows that allow them to browse through the available images and preview how they will appear in the document. End users can also double-click the image selector box to open the image selector window, which allows them to scroll through thumbnail versions of all available images. You can combine the capability to allow end users to edit image properties with the image selector. For example, you might want to allow end users to select an image, and then choose a border color that complements the image.

Keep in mind that if you want to allow end users to include images with transparency, the images must be in PNG format. Transparency in other image formats is not supported in LiveEditor. Also keep in mind that only certain output types support transparency in PNG images included in a Live document.

For more information about the output types that support transparency in PNG images, see [Importing External Content](#) in the Exstream Design and Production documentation.

For more information about how end users will use image selectors, see “[Using an Image Selector: The End User Experience](#)” on page 99.

4.3.1 Including Images as Part of the DLF File

You should include the images available in an image selector as part of the DLF file when you do not need to update the selection of images on a regular basis. Keep in mind, however, that this method creates larger files because the images are stored within the DLF itself.

For more information about DLF file structure and how objects are stored within a DLF, see “[A Technical Look at the DLF File Structure](#)” on page 27.

To set up an image selector that stores the available images within the DLF file, you must complete the following tasks:

- “[Placing an Image on the Page](#)” below
- “[Defining the Image Selector Properties for Images Included as Part of a DLF File](#)” on the next page
- “[Defining Other Image Properties](#)” on page 95

Note that only images from a local file system or mapped network drive can be used to create an image selector. You cannot use images from a connected repository.

Placing an Image on the Page

1. Open a design in Designer.
2. From the **Drawing Objects** toolbar, click .
3. Depending on whether you have an image repository configured and connected to Exstream, complete one of the following sets of steps on the **Import an Image** dialog box:

To	Do this
Insert an image from your local file system (if you do not have a repository set up)	<ol style="list-style-type: none">Select the image you want to place in the design.Click Open.

To	Do this
Insert an image from your local file system (if you have a repository set up)	 a. Click  b. Select the image you want to place in the design. c. Click Open .
Insert an image from a connected repository	Inserting images into a Live document from a connected repository is not currently supported.

4. To set the resolution of the image as it appears in the design, select one of the following options from the **Change resolution** drop-down list:

To	Do this
Maintain the original resolution settings of the image being imported	Select Keep as is .
Convert the image to a specific resolution	a. Select Convert to specified . b. In the New resolution box, enter a new resolution for the image.
Convert the image to the default design resolution specified for the design environment	Select Convert to design resolution .

Tip: For best results, images should be the same resolution as, or a multiple of, the resolution of the output.

5. To specify the number of bits used to save each pixel and the number of possible colors in the image, select the appropriate option from the **Color** drop-down list.
6. Click **OK** twice.

Defining the Image Selector Properties for Images Included as Part of a DLF File

1. In Designer, select the image you added and click .

The **Image Properties** dialog box opens.

2. Click the **Interactive** tab.

3. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.

The **Properties** check box and the **Use control to select content** drop-down list become available.

4. Select the **Properties** check box to let end users change properties for the object.

The **Image** tab, the **Lines and Fill** tab, and the **Placement** tab become available in LiveEditor.

5. From the **Use control to select content** drop-down list, select **Image selector**.

6. Click the **Image** tab.

7. In the **Selection variable** box, select a variable that will store the value of a selected image. You can use this variable in a formula to dynamically control other content on the page. For example, end users can select the appropriate logo for their branch. Based on that selection, the data about the branch location and hours can be added to the page automatically.

- a. Click .

The **Select Variable** dialog box opens.

- b. Navigate to the variable you want to use, select it, and click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Selection variable** box.

- c. If you specified an array variable, enter in the adjacent **Array element** box the element of the array that will store the value. Enter 0 if you want the element to be populated automatically.

8. Below the **Image selections** box, add another image to the image selector. You can add as many images as you want to the image selector.

- a. Below the **Image selections** box, click .

The **Import an Image** dialog box opens.

- b. Browse to the image you want to add.
- c. Click **Open**.

The **Import an Image** dialog box closes and the image appears in the **Image selections** box.

9. If you want to assign a value that is captured when an image is selected, complete the following steps:

- a. In the **Image selections** box, select an image.
 - b. In the **Selection value** box, enter a value for the image. You can use this value in formulas or other logic to drive processes or to update other values.
10. If you want to provide a description of an image to help end users pick the correct image, select the image in the **Image selections** box and enter descriptive information in the **Caption** box.
 11. In the **Image selections** box, select the image you want to appear on the page by default, and click the **Default selection** check box.
 12. From the Menu bar, select **Edit > Save**.

Defining Other Image Properties

You can set other properties for the image selector, such as border and placement properties. You define these properties the same way you set other properties on basic images.

For information about defining general image formatting properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

4.3.2 Dynamically Importing Images From a Shared Location

You can set up an image selector to import images dynamically from a shared location. This method lets you maintain images in a central location on a shared server instead of within the DLF file itself. If you maintain the images separately from the Live document, you can also change the value of a variable (by means of a rule, button, check box, and so on) to update images without having to generate a new DLF file.

For information about creating and changing variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

You can also dynamically control which images populate an image selector based on selections that end users make in a Live document. For example, suppose you design Live documents for a travel agency. You want all customer communications to contain pictures that are specific to each vacation destination. Each customer will receive a personalized letter with information about one of the destinations. However, you want to create only one Live document that contains image selectors with multiple image options for each destination. To set up this scenario, you could use variables and/or functions to associate a selection control (such as a drop-down list or selection group) in your Live document with dynamic image selector objects. In the Live document, when end users select a destination from the drop-down list or selection group, only images from that destination are available in the image selectors. If the end users change their selection, then only images from the new destination are available in the image selectors.

For information about setting up selection controls, see “[Controlling End User Editing Using Selection Controls](#)” on page 76.

For information about defining variables and functions, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To set up an image selector that dynamically imports images from a shared location, you must complete the following tasks:

1. “[Creating a New Placeholder Variable](#)” below
2. “[Assigning Values to the Placeholder Variable](#)” below
3. “[Creating an Empty Image Object for Dynamically Importing Images](#)” on the next page
4. “[Defining the Image Selector Properties for Dynamically Importing Images](#)” on the next page
5. “[Defining Other Image Properties](#)” on page 99

Creating a New Placeholder Variable

You must create a variable that defines the properties of the images that will be imported into the image selector. The placeholder must use the following settings:

- Its **Type** must be **Placeholder**.
- Its **Placeholder Type** must be **PNG**, **JPEG**, **TIFF (B&W G4 or uncomp)**, or **TIFF (Color)**.
- You must select the **Array** check box. If the placeholder variable is not an array, then the dynamic image selector will be empty when you create the Live document.

You can define the other variable properties as needed.

For information about setting up placeholder variables for Live documents, see “[Creating a Placeholder Document](#)” on page 368.

Assigning Values to the Placeholder Variable

After you have created a placeholder variable, you must assign to it the values that specify the location(s) of the images available in the image selector. You typically do this by mapping the fields in a data file to the newly created placeholder variable. You can also use the data file to specify the caption that appears with each image in the image selector window, as well as the selection value for each image.

For information about creating and mapping data files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

When assigning values to the placeholder variable, keep in mind the following behaviors:

- An error icon appears in the dynamic image selector window for each image specified by the placeholder variable that cannot be loaded into the dynamic image selector. Typically, end

users will see this error icon if an image is not available on a shared server, if the wrong name has been specified for the image, or if the image file is corrupt.

- If an end user selects an error icon from the dynamic image selector window, then the image object will be blank after the DLF has been processed.

Creating an Empty Image Object for Dynamically Importing Images

Unlike the process for setting up an image selector for images included in the DLF, when you set up a dynamic image selector, you must use an empty image object. An empty image object contains the images available to the end users (as specified by the placeholder variable).

To create an empty image object for dynamically importing images:

1. In Designer, on the Drawing Objects toolbar, click  .

The **Create an empty image** dialog box opens.

2. In the **Placeholder variable** box, select the placeholder variable that you created earlier.

- a. Click  .

The **Select Variable** dialog box opens.

- b. Navigate to the placeholder variable you want to use, select it, and click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Placeholder variable** box.

3. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.

4. From the **Use control to select content** drop-down list, select **Image selector**.

5. Click **OK**.

The **Create an empty image** dialog box closes and the placeholder area appears on the page.

6. Change the size and location of the empty image object as needed.

7. From the Menu bar, select **Edit > Save**.

Defining the Image Selector Properties for Dynamically Importing Images

1. In Designer, select the image you added and click .

The **Image Properties** dialog box opens.
2. Click the **Interactive** tab.
3. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.

The **Properties** check box and the **Use control to select content** drop-down list become available.
4. To allow end users to change properties for the object, select the **Properties** check box.

The **Image** tab, the **Lines and Fill** tab, and the **Placement** tab become available in LiveEditor.
5. From the **Use control to select content** drop-down list, select **Image selector**.
6. Click the **Image** tab.
7. In the **Selection variable** box, in the **Selector properties** section, select a variable that will store the value of selected images. This will not be the placeholder variable that you created earlier. Instead, you can use this variable in a formula to dynamically control other content on the page. For example, end users can select the appropriate logo for their branch. Based on that selection, the data about the branch location and hours can be added to the page automatically. Each image specified in a dynamic image selector must have its own value in the selection variable. If an end user selects an image that does not have a value in the selection variable, then the image object will be blank after the DLF has been processed.
 - a. Click .

The **Select Variable** dialog box opens.

 - b. Navigate to the variable you want to use, select it, and click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Selection variable** box.

When setting up a placeholder variable to populate an image selector object, you must specify a selection variable. If you do not specify a selection variable, the dynamic image selector will be empty when you create the Live document.
8. In the **Array containing selection values** box, specify a variable that provides the selection values for the images in the image selector.

- a. In the **Array containing selection values** box, click .

The **Select Variable** dialog box opens.

- b. Select the variable that provides the selection values and click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Array containing selection values** box.

9. If you want to provide a description for each image to help end users pick an image in the image selector window, select the appropriate array variable in the **Array containing caption values** box. Keep in mind that this variable can be mapped to the same data file that specifies the location and selection values for the images.

- a. Click .

The **Select Variable** dialog box opens.

- b. Navigate to the variable you want to use, select it, and click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Array containing caption values** box.

10. From the Menu bar, select **Edit > Save**.

Defining Other Image Properties

You can set other properties for the image selector, such as border and placement properties. You define these properties the same way you set other properties on basic images.

For information about defining general image formatting properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

4.3.3 Using an Image Selector: The End User Experience

This section describes how end users will use image selectors in LiveEditor.

1. In a Live document, end users will select the image that they want to change.
2. The image selector icon and arrows appear next to the image.
3. End users can either double-click the image selector icon to open a widow that displays all of the possible images, or they can click the arrows to browse through the images.

Image selector window



End users can use the image selector to change the image at a later time if they need to select a different image. Also, if you have allowed end users to change the properties of the image, such as its border, they can make those changes after they select the image from the image selector.

For information about allowing end users to change properties of images, see *Designing Customer Communications* in the Exstream Design and Production documentation.

4.4 Using a Content Selection Group to Dynamically Select Content

Content selection groups are selection control objects that allow end users to select the content to be included in a Live document and see how the Live document's appearance changes with the inclusion or exclusion of specific content. Content selection groups leverage the functionality of check boxes and radio buttons to make it easy for end users to select entire paragraphs of content for inclusion or exclusion. In addition, when you set up a content selection group, a dialog box is automatically created that contains the options an end user can select. End users can either select the check boxes and radio buttons next to paragraphs on the page,

or they can make selections in the dialog box to indicate which paragraphs should be included in the Live document.

For an overview of how the end users will use a content selection group, see “[Using a Content Selection Group: The End User Experience](#)” on page 105.

You can also set up content selection groups so that a selection is made automatically, based on the data an end user provides or other action. These types of content selection groups are called “system selection groups.”

Tip: You can use the `LiveSelection` built-in function to open the content selection group dialog box automatically based on an end user action or other event in LiveEditor.

For information about the `LiveSelection` built-in function, see “[LiveSelection](#)” on page 413.

By default, activation buttons appear near content selection groups on the design to help indicate the content selection group’s presence to end users. You can control whether the activation button appears for content selection groups.

For information about controlling whether activation buttons appear for content selection groups, see “[Controlling the Visibility of Activation Buttons in LiveEditor](#)” on page 120.

To use a content selection group to let end users make selections from a group of objects, you must complete the following tasks:

1. “[Creating a Variable to Control the Content Selection Group](#)” below
2. “[Creating the Objects to be Used in the Content Selection Group](#)” on the next page
3. “[Defining the Content Selection Group Settings](#)” on page 103

You can also complete the following optional task as needed:

- “[Locating Content Selection Group Variables](#)” on page 105

4.4.1 Creating a Variable to Control the Content Selection Group

Before defining the content selection group settings in Designer, you must create and define a content selection group variable. Content selection group variables are array variables that you configure to store or control the selection of objects in the content selection group.

If you are creating a system selection group (that is, if you are using data to automatically select content in the content selection group), you must also define logic that specifies the value of the variable.

For more information about using logic to specify the value of a variable, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

To create a variable to control the content selection group:

1. In Design Manager, create a new array variable.
For information about creating variables, see the *Using Data to Drive an Application* in the Exstream Design and Production documentation.
2. From the Library, drag the variable to the Property Panel.
3. Click the **Interactive** tab.
4. From the **Controls Live selection group** drop-down list, complete one of the following sets of steps:

To	Do this
Allow end users to select one or zero of the items in the content selection group	Select Pick one (Optional) .
Force end users to select one of the items in the content selection group	Select Pick one (Required) .
Force end users to select a specified number of items from the content selection group	<ol style="list-style-type: none">a. Select Pick fixed number.b. In the Number of selections allowed box, enter the number of selections an end user must make.
Allow end users to select a specified range of items from the content selection group	<ol style="list-style-type: none">a. Select Pick variable number.b. In the Number of selections allowed box, enter the range of selections an end user must make.

5. From the Menu bar, select **File > Save**.

4.4.2 Creating the Objects to be Used in the Content Selection Group

Each item in a content selection group can be either a textual paragraph or an entire text box, and you design these objects as you normally would. For example, suppose you are using a content selection group to allow end users to add pre-written paragraphs to a letter. You might create one text box to contain four separate paragraphs that end users can choose from. When you set up the content selection group, you will define the properties of each paragraph in such a way that they can be selected as separate items in the group.

As you design the objects that will make up the content selection group, keep in mind the following considerations:

- You do not need to set up check boxes or radio buttons that correspond to each item. Check boxes or radio buttons (depending on whether you allow end users to select one or multiple items) are added automatically, when you define the content selection group settings.

- You can combine the capability to allow end users to edit text with the content selection group. For example, you might want to include in the content selection group a paragraph that end users can edit and format to provide further customization in the Live document.
- The visibility of content selection groups in LiveEditor is controlled by the **Initial behavior of hidden items** property on the setting object. If you want content selection groups to be visible when the Live document is opened, select **Show** from the **Initial behavior of hidden items** drop-down list.

For more information about hiding and showing objects in LiveEditor, see “[Using Show/Hide to Simplify Editing](#)” on page 135.

- You can specify the font and font size used in the selection group content that appears in the dialog box on **Fonts to use in LiveEditor controls** property.

For information about customizing the fonts used in LiveEditor controls, see “[Changing the Fonts Used in LiveEditor Controls](#)” on page 121.

4.4.3 Defining the Content Selection Group Settings

1. In Designer, select the first object that makes up the content selection group:

To	Do this
Use a text paragraph in the content selection group	Right-click the text paragraph and select Text paragraph properties .
Use a text box in the content selection group	Select the text box and click  .

The object properties dialog box opens.

2. Click the **Interactive** tab.
3. In the **Show and hide** area, from the drop-down list, select one of the following options:

To	Do this
Set up a content selection group in which selected objects appear based on end users' selections	Select Selection group - user may pick .
Set up a content selection group in which selected objects appear automatically, based on customer data	Select Selection group - system only .

4. Use the **Selection group variable** box and **Value** box to specify the variable that controls the content selection group or the value that is stored when an end user selects the current item.

- a. In the **Selection group variable** box, click  .
The **Select Variable** dialog box opens.
- b. Select the variable that stores the selection in the selection group (if you selected **Selection group - user may pick** from the **Show and hide** drop-down list), or select the variable that controls the content selection group (if you selected **Selection group - system only** from the **Show and hide** drop-down list), and click **OK**.

The **Select Variable** dialog box closes and the variable that you selected appears in the **Selection group variable** box.

- c. In the **Value** box, enter one of the following values, depending on the value that you selected from the **Show and hide** drop-down list:

Selected value	Enter this value
Selection group - user may pick	The value that is stored when the user selects this object in the content selection group
Selection group - system only	The value of the variable for which this object is automatically selected in the content selection group

5. If you selected **Selection group - user may pick**, select one of the following options from the **Selection prompt** drop-down list to specify the text that appears in the dialog box when end users click the content selection group:

- **Edit area name**—The text that appears in the **Name** box on the **Advanced Properties** dialog box. Use this option when the object or text area is in a larger object, such as a paragraph in a text box.
- **Object name**—The text that appears in the **Reference name** box on the **Dynamic Size and Placement** tab. If you select this option and the content is located within a larger object, such as a text box, the text content is used instead.
- **Text content**—The content of an area of text or variable. Use this option if the content is short and can be easily read in the dialog box.
- **Live caption**—The text that appears in the **Live caption** box on the **Interactive** tab

6. Click **OK**.

The object properties dialog box closes and check boxes or radio buttons appear next to the content on the page.

7. Repeat step 1 through step 6 for each object that you want to appear as an option in the content selection group.
8. From the Menu bar, select **Edit > Save**.

4.4.4 Locating Content Selection Group Variables

Sometimes, you might need to quickly determine which variable is used to control a specific content selection group. For example, if you want to use the content selection group variable in a formula to update other areas of the Live document, you can use the search functionality to locate all content selection group variables on a page and identify the content selection group with which each variable is associated.

To locate content selection group variables in Designer, select **Edit > Search > Live Selection Groups** from the Menu bar. The **Selection groups** dialog box opens and displays all of the content selection group variables located on the page.

Content selection groups dialog box

Name	# Pick	Description
Claim_DocumentAttachments	1 - 3	Select any number
Claim_CustomerFollowUp	Pick 1	Select one

You can double-click a variable in the **Selection groups** dialog box to automatically highlight the content selection group it controls.

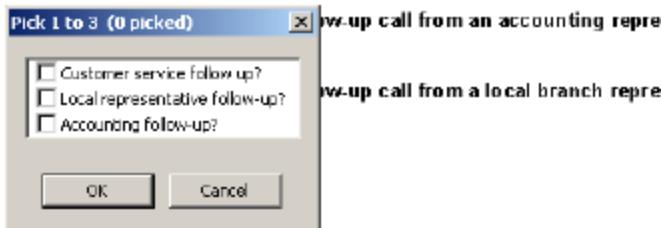
4.4.5 Using a Content Selection Group: The End User Experience

This section describes how end users will use content selection groups in LiveEditor.

In a Live document, end users will see a list of options from which they can make selections. End users can either select a radio button or check box to make a selection, or they can click the content itself to open a dialog box that displays all of the possible selections. Either method allows them to make the number of selections allowed for that particular content selection group.

Example of a content selection group

This customer requires a follow-up call from a customer service or enrollment process or other specific questions.



After end users make a selection from either the content selection group or the content selection group dialog box, the selection group variable is updated with the value that you specified for the user's selection, and the Live document is updated according to the logic that you designed. End users can change their selections as needed.

4.5 Using Buttons to Launch Actions

Exstream Live allows you to create buttons that are customized for a particular environment or document. When a document containing a button is opened in LiveEditor, end users can click the button and the specified action occurs. For example, a common use of buttons is to "hide" complex processes such as submitting the content of a Live document to a web service or updating a document with changed data. You can also use buttons for simpler actions, such as running a basic built-in function, like LiveSave.

When you use buttons in a Live document, you can associate external hyperlinks or functions with a button. You can also customize the appearance of a button by providing an image that appears in place of a traditional button. The following example illustrates two different types of buttons that might appear at the top of a Live document: an image used as a button to take end users to the previous page, and a traditional button that allows end users to submit data to their content repository.

Example of button types



As with other selection controls, you must specify the variable that will be populated when the button is selected, and how the value of that variable is stored.

To set up a button, you must complete the following tasks:

1. “[Adding a Button to a Page](#)” below
2. “[Defining the Variable Settings of the Button](#)” below
3. “[Customizing the Appearance of the Button](#)” on the next page
4. “[Specifying the Button Actions](#)” on page 110

4.5.1 Adding a Button to a Page

In Designer, on the Drawing Objects toolbar, click . A button is placed on the page. You can use the tools in Designer to move and resize the button as needed.

4.5.2 Defining the Variable Settings of the Button

You use the options in the variable settings area of the button properties to specify the variable to be populated with a value when the button is selected, and how the value of that variable is stored.

Before beginning this task, you must have created a variable designed to store the value of the button when it is selected.

For more information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To define the variable settings for a button:

1. In Designer, select the button and click .
- The **Button Properties** dialog box opens.
2. Click the **Button** tab.
3. Use the **Variable** box to specify the variable that will be populated with a value when an end users selects the button. The variable should be a Boolean, string or integer variable.

- a. Click .

The **Select Variable** dialog box opens.

- b. Select the variable and click **OK**.

The **Select Variable** dialog box closes and the variable you selected appears in the **Variable** box.

4. If you specified an array in the **Variable** box, enter the index value of the array element that will be populated when the radio button is selected in the **Array element** box. Enter 0 if you

want the element to be populated automatically.

5. Use either the **Value if clicked** box or the **Array containing selection values** box to specify the value stored when the radio button is selected.

To	Do this
Store a static specified value	In the Value if clicked box, enter the value you want to populate the specified variable when the button is selected. This box is active only if you specified a string or integer variable.
Store a variable-driven value	a. In the Array containing selection values box, click  . The Select Variable dialog box opens. b. Select the variable that provides the values and click OK . The Select Variable dialog box closes and the variable you selected appears in the Array containing selection values box.

6. From the Menu bar, select **Edit > Save**.

If you use an array variable to store the value of the button, and the button is placed in an object that can repeat dynamically (such as a table row), the elements in the array are populated with values in the order that the buttons appear. For example, if the button is located in a repeating table row, the first element in the array contains the value for the radio button in the first row, the second element contains the value for the button in the second row, and so on.

If you use array variables to set up the button values, and the button is placed in an automated data-section driven table, all of the variables used to define the button must be associated with the data section. Therefore, if you cannot map the variable that stores the button value in the data section, you must use the VARSCOPE switch in a control file to specify the section that is associated with the variable.

For more information about the VARSCOPE switch, see *Switch Reference* in the Exstream Design and Production documentation.

4.5.3 Customizing the Appearance of the Button

You can customize buttons by changing the text used on the button label, or by providing an image that appears in place of a traditional button.

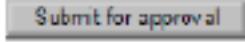
Note: LiveEditor does not support the use of complex text languages (such as Arabic, Farsi, or Hebrew) to customize the text that appears on a button. If you are using a complex text language to customize a button, you must first create an image that contains the complex text, and then use the image for the custom button.

To customize the appearance of a button:

1. In Designer, select the button and click .

The **Button Properties** dialog box opens.

2. Click the **Button** tab.
3. Use the **Button style** and **Toggle** options to specify the appearance of the button:

To	Example	Do this
Provide custom text that appears on the button		<ol style="list-style-type: none">From the Button style drop-down list, select Drawn (button).If you want to allow the button to "remember" its state of being clicked or unclicked, select the Toggle check box. The toggle feature also allows you to provide different text that appears on the button depending on whether it has been clicked or not. (For example, if you include several sections in a form that require end users to click a button after each section to indicate that the section is complete, using a toggle button lets end users quickly see which sections have been completed.)In the Caption box, enter the text you want to appear on the button. If you selected the Toggle check box, this text will appear before the button is clicked. If you are using a complex text language, you cannot use this setting; instead, you must use an image that contains the complex text. For instructions on using an image for a custom button, see "Use an image you select" on the next page.If you selected the Toggle check box, enter in the Down caption box the text that appears on the button when an end user clicks the button (in other words, when the button is placed into a "down" state). For example, you might enter Submit as the caption text and Submitted as the down caption text. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"><p>Note: If you assign a variable to a toggle button, the variable sets to the Value if clicked value when an end user clicks the button (placing the button into the "down" state). If an end user clicks the button again (returning it to the "up" state), the value of the variable resets to the default value.</p></div>

To	Example	Do this
Use an image you select	Varies	<p>Note: Only images from a local file system or mapped network drive can be used to customize buttons. You cannot use images from a connected repository.</p> <p>Important: If you want to use a complex text language (such as Arabic, Farsi, or Hebrew) to customize a button, you must use this option. You must first create an image that contains the complex text, and then use the image for the custom button.</p> <ol style="list-style-type: none">From the Button style drop-down list, select Images. Additional options appear at the bottom of the dialog box.Double-click the Up box. The Import an Image dialog box opens.Browse to and select the image that you want to be used for the button when it has not been clicked.Click Open. The Import an Image dialog box closes and the image you selected appears in the Up box.Repeat step b through step d for the Down box (image when the button has been selected) and Hover box (image when an end user hovers the pointer over the button).

4. Click **OK**.

The **Button Properties** dialog box closes.

5. From the Menu bar, select **Edit > Save**.

4.5.4 Specifying the Button Actions

When an end user clicks a button, an action is triggered. This action can either be a function that is executed or a link that is followed.

Before beginning this task, you must have created a function designed to carry out the intended action of the button.

For information about creating functions, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

For information about Live built-in functions, see “[Live Built-In Functions](#)” on page 381.

To specify the action to be triggered when a button is clicked:

1. In Designer, select the button and click .

The **Button Properties** dialog box opens.

2. Click the **Interactive** tab.

3. Click .

The **Advanced properties** dialog box opens.

4. Use the options on the dialog box to specify the button actions:

To	Do this
Follow a link when the button is clicked	<p>From the Link drop-down list, select Static if you want to enter a static link that is followed or select Variable if you want to provide a variable that supplies the link.</p> <ul style="list-style-type: none">• If you selected Static, enter the link in the box below the Link drop-down list.• If you selected Variable, click  in the variable box and select the variable that provides the link values.
Trigger a function when the button is clicked	<ol style="list-style-type: none">a. In the Button action box, click  . The Select Function dialog box opens.b. Select the function that provides the action you want to launch when the button is clicked and click OK. The Select Function dialog box closes and the function you selected appears in the Button action box.

5. Click **OK**.

The **Advanced properties** dialog box closes.

6. Click **OK**.

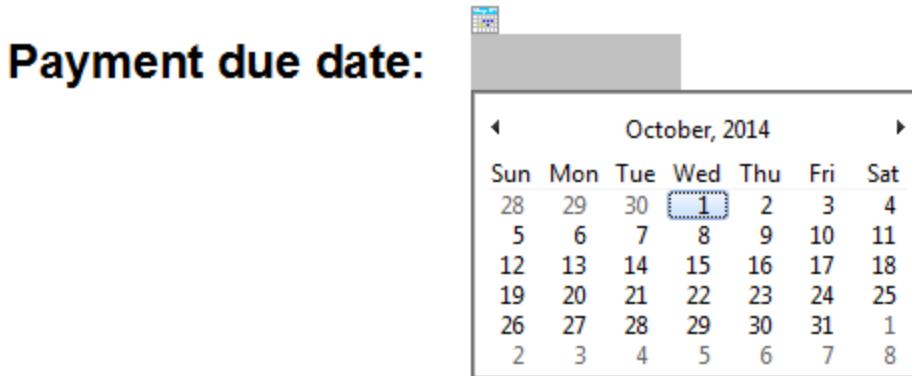
The **Button Properties** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

4.6 Using a Calendar to Allow End Users to Choose Dates

You can use a calendar object to make it easy for end users to select dates that appear at a given location. Calendars appear as a graphical representation of a printed calendar, and end users can browse through the months and years to select a date. For example, calendars are often used when you make date variables editable. End users can select a date from a calendar, and the value of a date variable is updated automatically.

Example of calendar



If you use a calendar to help end users change the value of a variable, the date end users select is formatted according to the option selected on the **Output format** drop-down list of the variable properties. If the calendar is not tied to a variable, the selected date is formatted according to the option selected from the **Date format** drop-down list on the locale used for the application.

For information about setting the output format for variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

For information about locales, see *System Administration* in the Exstream Design and Production documentation.

By default, activation buttons appear near calendars on the design to help indicate the calendar's presence to end users. You can control whether the activation button appears for calendars. You can also control the fonts used in the calendar.

For information about controlling whether activation buttons appear for calendars, see [“Controlling the Visibility of Activation Buttons in LiveEditor” on page 120](#).

For information about changing the fonts used in a calendar, see [“Changing the Fonts Used in LiveEditor Controls” on page 121](#).

To use a calendar to help end users choose dates:

1. In Designer, do one of the following to select the text or variable where you want end users to specify a date and open its properties:

To	Do this
Allow end users to use a calendar to replace an area of text	<ol style="list-style-type: none">Select the lines or paragraphs of text.Right-click the selected text and select Text paragraph properties. The Text paragraph properties dialog box opens.

To	Do this
Allow end users to use a calendar to replace all of the content in a text box	<ol style="list-style-type: none">a. Select the text box.b. Click  . <p>The Text Properties dialog box opens.</p>
Allow end users to use a calendar to update a variable value	Right-click the variable and select Variable properties . The Variable Properties dialog box opens.

2. Click the **Interactive** tab.
3. From the **Content change** drop-down list, select either **Change is optional** or **Change is required**, as needed.
Additional options become available.
4. Select the **Text edit** check box.
5. From the **Use control to select content** drop-down list, select **Calendar**.
6. Click **OK**.
The properties dialog box closes.
7. From the Menu bar, select **Edit > Save**.

When end users select the text or variable in LiveEditor, the  button appears. When the end users click , the calendar opens and they can make a selection from it.

4.7 Using Form Fields to Control Data Entry

Form fields are objects that you can use to control the type and amount of data that an end user can enter in a given area. For example, suppose end users must enter a customer's phone number in a Live document. When the Live document is processed, the phone number will be collected and stored in your organization's database. If you simply enable free-form text editing, an end user might enter the phone number in an invalid format or accidentally press a letter instead of a number. However, if you use a form field, you can automatically format data entered in the area to comply with your data entry requirements and reject any characters that are not numbers from being entered as part of the phone number. Form fields are often used when you allow end users to change the value of a variable, since form fields can help ensure that the new variable value conforms to the expected format and content.

For information about validating data, see “[Validating Data Entered by End Users](#)” on page 284.

Example of form field

Zip code:

In Live documents, form field objects are made up of two components: the properties that define the physical appearance of the form field, and the characters that control the type of data that can be entered in the form field (these characters are called a data entry mask). The physical properties and the data entry mask are defined and controlled separately, giving you flexibility in the way form fields are defined, and allowing reuse between form field objects. For example, you can set up form fields in which only the properties that control the appearance of the form field are defined. Then, as the form field is added to pages in Designer, the data entry mask for each instance of the frame is customized for the type of data that will be entered (such as an account number, Social Security number, or date of birth). Although setting up pre-made form field objects can save you time as you design a Live document, they are not required. If you rarely use form fields in your design, or if you want to format each instance of a form field differently, you might choose to create a new form field each time you use one.

This section discusses the following topics:

- [“Setting Up Pre-Made Form Fields” below](#)
- [“Adding Form Fields to a Page” on page 116](#)
- [“Data Entry Mask Symbols” on page 118](#)

4.7.1 Setting Up Pre-Made Form Fields

You can set up pre-made form fields that you can then add to a page in Designer. For example, if you want to use the colors in a form field to help provide visual guidance to end users about the type of data that should be entered, you might set up three default form fields: a black form field for phone numbers, a red form field for Social Security numbers, and a blue form field for account numbers. These pre-made form fields appear in the Library in Design Manager, and can be edited and managed just like other Library objects. Therefore, these types of form fields are sometimes called “Library form fields.” When you design a page in Designer, the form fields will appear as options when you add a form field to the page.

In addition to setting up pre-made Library form fields, you can also create an application default form field. An application default form field is a Library form field that is selected as the default form field to be used throughout a Live document application. You might choose to specify an application default form field if you have designed one form field that should be used in almost every form field instance.

This section discusses the following topics:

- [“Creating Library Form Fields” on the next page](#)
- [“Specifying a Form Field as the Application Default Form Field” on page 116](#)

Creating Library Form Fields

1. In Design Manager, in the Library, go to **Environment > Interactive > Form Fields**.
2. Right-click the **Form Fields** heading and select **New Form Field**.
The **New Form Field** dialog box opens.
3. In the **Name** box, enter a name. In the **Description** box, enter a description (optional).
4. Click **Finish**.

The **New Form Field** dialog box closes and the form field appears in the Property Panel for you to define.

5. From the **Character frame** drop-down list, select one of the options to specify the type of border you want to appear around each character in the form field:

Character frame options

Option	Example
None	0 1 2 3 4 5
Box	0 1 2 3 4 5
Bevel	0 1 2 3 4 5
Comb	0, 1, 2, 3, 4, 5
Comb (top and bottom)	0 1 2 3 4 5
Underscore	0 _ 1 _ 2 _ 3 _ 4 _ 5
Corners	0 ↗ 1 ↗ 2 ↗ 3 ↗ 4 ↗ 5 ↗

6. Use the **Line** color well and box to define the color and thickness of the lines that appear around each character in the form field.
7. Use the **Fill** color well to define the fill color and pattern that appears behind each character in the form field. This color appears when an end user enters characters in the field.
8. By default, the size of the frame that appears around each character in the form field is determined automatically, based on the size of the character. If you want to manually specify the size of the character frames, clear the **Autosize** check box and use the

Spacing, **Height**, and **Width** boxes below to specify the frame size.

9. From the Menu bar, select **File > Save**.

Specifying a Form Field as the Application Default Form Field

1. In Design Manager, from the Library, drag the application to the Property Panel.
2. Click the **Interactive** tab.
3. In the **Default form field** box, click .
The **Select Form Field** dialog box opens and displays all of the existing form fields.
4. Select the form field that you want to serve as the application default form field and click **OK**.
The **Select Form Field** dialog box closes and the form field you selected appears in the **Default form field** box.
5. From the Menu bar, select **File > Save**.

4.7.2 Adding Form Fields to a Page

1. In Designer, do one of the following to select the text that you want to allow end users to change using a form field:

To	Do this
Allow end users to format of all the text within a text box	<ol style="list-style-type: none">a. Select the text box.b. Click . <p>The Text Properties dialog box opens.</p>
Allow end users to format only a specific paragraph of text in a text box or table cell	<ol style="list-style-type: none">a. Select the lines or paragraphs of text.b. Right-click the selected text and select Text paragraph properties. <p>The Text paragraph properties dialog box opens.</p>

2. Click the **Interactive** tab.
3. From the **Content change** drop down list, select either **Change is optional** or **Change is required**, as needed.

Additional options become available.

4. Select the **Text edit** check box.

Additional options become available.

5. From the adjacent drop-down list, select **Form field**.

The  button becomes available.

6. Click .

The **Field Properties** dialog box opens.

7. From the **Form field source** drop-down list, select the type of form field you want to add to the page:

To	Do this
Use the application default form field (if one is created) For information about application default form fields, see " Specifying a Form Field as the Application Default Form Field " on the previous page.	Select Application default .
Use a pre-made Library form field For information about creating Library form fields, see " Creating Library Form Fields " on page 115	<ol style="list-style-type: none">a. Select Library form field.b. In the Library form field box, click . <p>The Select Form Field dialog box opens and displays all of the existing form fields.</p> <ol style="list-style-type: none">c. Select the form field you want to use and click OK. <p>The Select Form Field dialog box closes and the form field you selected appears in the Library Form Field box.</p>
Create a new form field	<ol style="list-style-type: none">a. Select Local form field. <p>The Form field properties area becomes active.</p> <ol style="list-style-type: none">b. From the Character frame drop-down list, select the type of border you want to appear around each character in the form field.c. Use the Line color well and box to define the color and thickness of the lines that appear around each character in the form field.d. Use the Fill color well to define the fill color and pattern that appears behind each character in the form field. This color appears when an end user enters characters in the field.e. By default, the size of the frame that appears around each character in the form field is determined automatically, based on the size of the character. If you want to manually specify the size of the character frames, clear the Autosize check box and use the Spacing, Height, and Width boxes below to specify the frame size.

8. In the **Default data entry mask** box, enter the combination of characters and symbols that defines the type of data that can be entered in the form field. For example, to force the data

to conform to the format of a telephone number using the North American Numbering Plan, enter **###-##-####**.

For information about the symbols that you can use to create the data entry mask, see [“Data Entry Mask Symbols” below](#).

9. Click **OK**.

The **Field Properties** dialog box closes.

10. Click **OK**.

The object properties dialog box closes.

11. From the Menu bar, select **Edit > Save**.

4.7.3 Data Entry Mask Symbols

The following table lists the types of symbols you can use in a data entry mask:

Data entry mask symbols

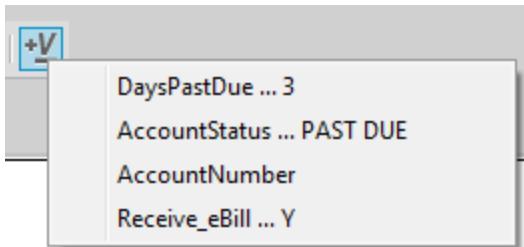
Symbol	Meaning
*	Any character
#	A numeric character (0–9)
?	An alphabetic character (A–Z, a–z)
A	An alphanumeric character (0–9, A–Z, a–z)
U	An uppercase character (A–Z)
L	A lowercase character (a–z)
H	A hexadecimal digit (0–9, A–F, a–f)
\	Removes the symbol designation from the following character

If you enter a character other than one of the designated symbols, the data entry mask automatically populates the text with that character. For example, if you create the **##-##-####** data entry mask, the dashes are automatically included in the form field.

4.8 Setting Up a Variable Palette to Allow End Users to Insert Variables

One advanced feature of Exstream Live is the ability to allow end users to add variables to a Live document. By default, this functionality is disabled, and end users cannot add variables to editable areas in a Live document. However, by specifying variables that can appear in a Variable Palette in LiveEditor, you can enable end users to select and include one of those variables in any area where they can edit text.

Example of LiveEditor Variable Palette



Tip: Because variables are complex objects and end users do not have all of the information about the variables in LiveEditor, enable this functionality thoughtfully and give end users as much assistance as possible in choosing the appropriate variable.

To set up a Variable Palette to allow end users to insert variables:

1. In Design Manager, from the Library, drag the application to the Property Panel.
2. Click the **Interactive** tab.
3. Use the **Variables to include on the Variable Palette in Live** and the **Variable Selector in Empower** properties to complete one of the following sets of steps:

To	Do this
Automatically include all of the variables used in the Live document on the Variable Palette	From the drop-down list, select All .

To	Do this
Specify individual variables to appear on the interactive Variable Palette	<ol style="list-style-type: none">a. From the drop-down list, select Listed.b. Click  . The Select Variable dialog box opens.c. Select the first variable you want to include on the variable palette and click OK. The Select Variable dialog box closes and the variable you selected appear in the box.d. Repeat step b through step c to add as many variables as you want.

4. From the Menu bar, select **File > Save**.

4.9 Controlling the Visibility of Activation Buttons in LiveEditor

Some selection control objects have activation buttons that help indicate to end users that a selection must be made in a given area. Activation buttons are small graphical indicators that appear just above editable areas in a Live document. When an end user clicks the activation button, the control opens.

Showing or hiding activation buttons can help end users better navigate through Live documents. For example, if you create a form that contains many calendars and/or drop-down lists, you can hide the activation buttons for each of those objects to reduce the amount of visual clutter on the page. On the other hand, if you create a Live document that contains only a few calendars and/or drop-down lists, you can show activation buttons for each so that end users can more quickly recognize the interactive areas that require their input. Keep in mind, however, that the changing the visibility of activation buttons in a Live document does not force end users to make selections in a given area; you must enforce that through the object properties.

You can control whether end users see activation buttons in LiveEditor for calendars, drop-down lists, selection groups, and objects that can be hidden.

You can control the visibility of the following activation buttons:

Interactive area	Activation button(s)
Calendars	
Drop-down lists	

Interactive area	Activation button(s)
Content selection groups	 or 
Objects that can be hidden	 or 

Keep in mind that end users can access a menu option in LiveEditor (by selecting **Live > Show activation buttons** from the Menu bar) that shows or hides activation buttons in a Live document. Therefore, even if you set up a Live document so that the activation buttons are not visible, end users can still make them visible from within LiveEditor.

To specify the visibility of activation buttons in a Live document:

1. In Design Manager, from the Library, drag the setting object you will use to create the Live document to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Initial behavior of Interactive area buttons** drop-down list, select one of the following options:
 - **Hide**—Hides activation buttons in the Live document. **Hide** is the default setting.
 - **Show**—Shows activation buttons in the Live document
4. From the Menu bar, select **File > Save**.

4.10 Changing the Fonts Used in LiveEditor Controls

You can control the font that is displayed in selection controls in LiveEditor, such as calendar controls, in order to match the look and feel of the rest of the Live document. You can select from fonts that you have installed on your system. However, you must make sure end users have the proper fonts set up on their computer. If an end user does not have the selected font installed, the operating system will use the font that most closely matches the selected font. The default font is Arial.

For information about setting up the fonts used in a Live application, see “[Controlling Font Behavior in Live Applications](#)” on page 248.

To change the font displayed in selection controls:

1. In Design Manager, from the Library, drag the setting object associated with the Live document to the Property Panel.
2. Click the **Editor Framework** tab.

3. From the **Font to use in LiveEditor controls** drop-down list, select the font that you want to use.

Note: This setting does not support complex text languages (such as Arabic, Farsi, or Hebrew).

4. From the **Size** drop-down list, select or enter the size that you want to use for the font.
5. From the Menu bar, select **File > Save**.

Chapter 5: Using Live Tools to Create a More Intuitive and Automated Editing Experience

As you move beyond creating basic Live documents, you can use various Live tools to make the editing experience more intuitive for end users and also automate repetitive tasks. Although these tools are more sophisticated from a design standpoint than those you might use to create basic Live documents, they allow you to streamline processes and simplify the way end users interact with LiveEditor. For example, suppose you create a long Live document that contains dozens of pages, but end users need to fill out only two or three form fields on each page. Instead of requiring end users to navigate through every page of the customer document, you can create an interview page that centralizes data entry while still populating all of the necessary form fields throughout the document.

This chapter discusses the following topics:

- “Using Interview Functionality to Drive Data Changes” below
- “Using Design Layers to Provide Instructions to End Users” on page 127
- “Using Actions to Automate Editing Tasks” on page 129
- “Executing Functions During LiveEditor Events” on page 132
- “Executing Functions During Editing” on page 134
- “Using Show/Hide to Simplify Editing” on page 135
- “Using External Hyperlinks to Link to Locations Outside of a Live Document” on page 137
- “Using Internal Hyperlinks to Link to Locations Within a Live Document” on page 149
- “Making Data Available in Live Documents” on page 150

5.1 Using Interview Functionality to Drive Data Changes

You can use interview functionality to streamline data entry, or to update variable data across a Live document. For example, you can use interview pages to simplify data entry so that the data that end users enter in a centralized location populates fields throughout a customer document. Likewise, you can use interview documents to dynamically update values for all customers included in a Live document.

This section discusses the following topics:

- “[Using Interview Documents in a Live Document](#)” below
- “[Using Interview Pages in a Live Document](#)” on the next page
- “[Designating a Variable to be an Interview Variable](#)” on page 126

5.1.1 Using Interview Documents in a Live Document

Interview documents are containers for pages containing data that is not customer-specific and has a single value across a set of customers. For example, suppose you are using Exstream Live to create a multi-customer brochure for a travel agency. The Live document contains three customer documents: one for customers interested in European vacations, one for customers interested in Asian vacations, and one for customers interested in African vacations. Most of the data used in each customer document is customer-specific, such as an address. However, each customer document also contains data that is common to all customers, such as the expiration date for a sale being advertised by the travel agency. Instead of requiring end users to change the common data in each customer document, you can set up an interview document that lets end users change common data in a single location but populates the data in each customer document. In that scenario, if an end user changes the expiration date on the interview document, then that date will show up on all three customer documents.

Creating an Interview Document

1. In Design Manager, from the Library, drag the Live application to the Property Panel.
2. Click the **Interactive** tab.
3. In the **Interview document** field, click . The **Select Document** dialog box opens.
4. Select a document from the list and click **OK**.
5. From the Menu bar, select **File > Save**.

As you add pages to an interview document, keep in mind that all pages in an interview document are considered interview pages even if you do not select the **Live interview page** check box on the page properties. Moreover, when you include an interview document in a Live document, it is shown at the top of the Outline Viewer.

5.1.2 Using Interview Pages in a Live Document

You can add interview pages to a Live document to make it easier for end users to enter and customize information. Unlike interview documents, which are used to update data across a set of customer documents, interview pages are typically used in on-demand situations when you need to customize a document based on a particular customer's information (for example, adding pages based on the customer's unique information). Usually, interview pages use questions and answers to gather the information needed to personalize a document for a particular customer.

For example, suppose you want to create a Live document with an interview page that is used by your customer service representatives. They can use the simple questions and check boxes to select what documents to send a caller. Documents are created and customized for the caller as the representative makes selections on the interview page.

Sample interview page in LiveEditor

Pecvniate obedivnt omnia

1. Si finis bonus est, totum bonum erit Salve Scio Semper
2. Omnim gathetur: Moore
3. Dic mihi solum facta, domina Dies Dixa Dulce
4. Margaritas ante porcos Margaritas Porcos

Tip: You can use design layers as an alternative to interview pages when you want end users to see what they are creating but you need to provide them with some additional information to guide them.

For information about design layers, see “[Using Design Layers to Provide Instructions to End Users](#)” on page 127.

Creating an Interview Page

1. In Design Manager, from the Library, drag the page to the Property Panel.
2. Click the **Basic** tab.
3. Select the **Live interview page** check box.

When you select the **Live interview page** check box, the **Use as a Web form** check box also becomes active. You can use this option to specify that the page will be transformed to XML during packaging so it can be presented as a web form.

For information about creating Live documents that can be presented as web forms, see “[Collecting Data For a Live Document Using Web Forms](#)” on page 197.

4. From the Menu bar, select **File > Save**.

When you have designated a page as a Live interview page, Exstream includes the page in the Live document so it can be viewed in LiveEditor. However, the page is not included in any other type of output. For example, if you produce output for both Live and AFP using the same application, the interview page is in the DLF output but not the AFP output. When an end user opens the interview page in LiveEditor, the visual indicators you specified for the interview page in your theme are used to designate it.

Tip: To print interview pages, use the PRINT_LIVE_INTERVIEW_PAGES engine switch.

For information about themes, see “[Using Themes to Visually Guide End Users](#)” on page 218.

5.1.3 Designating a Variable to be an Interview Variable

Interview variables are used in interview documents and have a single value that can be used across all customers in a Live document. This type of variable lets end users change values, such as a date, for all customers with a single change to a variable or other editing tool. For example, when you include the date variable as an interview variable, the date is updated for all customers when an end user changes it on the interview document.

This topic assumes that you have already created a variable. Using interview variables in your design allows end users to make a selection once and then values are updated based on their selection automatically throughout the document based on the selection.

For more information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To designate a variable to be an interview variable:

1. In Design Manager, from the Library, drag the variable to the Property Panel.
2. Click the **Interactive** tab.
3. Select the **Interview variable (unchanged across customers)** check box.
4. From the Menu bar, select **File > Save**.

5.2 Using Design Layers to Provide Instructions to End Users

Design layers are Exstream objects that let you provide guidance to end users as they work in different areas of a Live document. They serve a similar purpose in both Live and non-Live applications. However, while you might use design layers to represent non-printing objects (such as a pre-printed letterhead) in a non-Live document, you will typically use language layers to provide instructions to end users on a Live document.

Much like interview pages, design layers help you simplify the end user experience. However, interview pages tend to serve the purpose of centralizing data entry so that you can limit end user exposure to a Live document in its entirety. In contrast, design layers let you provide guidance to end users throughout a Live document. For example, suppose a page in your Live document contains an image selector object that allows end users to upload their own image. However, the image that end users upload must be a JPEG measuring 200 x 200 pixels. In that scenario, you can include a non-printing design layer that places a note beside the image selector object explaining the image requirements. Because the design layer is non-printing, it will not appear in the final output.

For more information about setting up design layers, see *Designing Customer Communications* in the Exstream Design and Production documentation.

This section discusses the following topics:

- “[Controlling the Visibility of Design Layers in a Live Document](#)” below
- “[Controlling Whether Design Layers Are Printed When Producing Output from a Live Document](#)” on the next page
- “[Disabling All Design Layers](#)” on page 129

5.2.1 Controlling the Visibility of Design Layers in a Live Document

You can specify when/if a design layer is visible on a Live document, and whether end users can control its visibility.

This topic assumes that you have already created a design layer.

For more information about creating design layers, see *Designing Customer Communications* in the Exstream Design and Production documentation.

To specify how design layers behave when viewing a Live document:

1. In Design Manager, from the Library, drag the design layer to the Property Panel.
2. From the **Interactive visibility** drop-down list, select one of the following options:
 - **Always**—The design layer is always visible and end users cannot hide it.
 - **Never**—The design layer is never visible and end users cannot show it.
 - **User selectable**—End users can turn the design layer off and on using the Design Layer panel.
 - **Live toggle**—End users can use the button to turn all the design layers in the Live document off and on.
3. From the Menu bar, select **File > Save**.

5.2.2 Controlling Whether Design Layers Are Printed When Producing Output from a Live Document

You can control whether a design layer will show up in printed output when an end user prints a Live document. For example, you might want a design layer that contains the text "DRAFT" to show up on output only until a Live document is submitted for processing. In that scenario, you can create a rule that suppresses the design layer after the Live document has been submitted.

This topic assumes that you have already created a design layer.

For more information about creating design layers, see *Designing Customer Communications* in the Exstream Design and Production documentation.

To specify how design layers behave when viewing a Live document:

1. In Design Manager, from the Library, drag the design layer to the Property Panel.
2. From the **Live print visibility** drop-down list, select one of the following options:
 - **Always**—The design layer always prints.
 - **Never**—The design layer never prints. When you select this option, the objects on the design layer do not appear on local print events.
3. From the Menu bar, select **File > Save**.

5.2.3 Disabling All Design Layers

If you do not want design layers to appear in a Live document, you can suppress them globally. Keep in mind, however, that this setting on the Live setting object overrides any visibility settings that you specify on the design layers themselves.

To disable all design layers in a Live document:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Design** tab.
3. Select the **Disable design layers** check box.
4. From the Menu bar, select **File > Save**.

5.3 Using Actions to Automate Editing Tasks

Actions allow you to make custom activities easier by building them into a Live document and making them available to end users in LiveEditor. For example, you can make it easier for your end users to contact your support team by creating an action to open an email program, supply the address for your support team, and attach the DLF file being edited.

To apply an action to a Live document, you must first include the action in a Live setting object. You then associate the Live setting object with a DLF output object. When you create the DLF output, the Live setting object (and therefore the action) is applied.

For more information about DLF output objects and Live setting objects, see “[Setting Up the Objects Necessary for Live Output](#)” on page 354.

This section discusses the following topics:

- [“Setting Up an Action” below](#)
- [“Specifying an Icon to Display in a Toolbar” on page 131](#)
- [“Adding Actions to Appear on the Live Menu” on page 132](#)

5.3.1 Setting Up an Action

Before creating a new action, you must have defined a function that will be executed by the action.

To create an action:

1. In Design Manager, in the Library, go to **Environment > Interactive > Actions**.

2. Right-click the **Actions** heading and select **New Action**.

The **New Action** dialog box opens.

3. In the **Name** box, enter a name. In the **Description** box, enter a description (optional).

4. Select the **Associated function**.

Note: You must select a function to continue. You can change the **Associated function** on the Property Panel.

For more information about functions, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

5. Click **Finish**.

The new action opens in the Property Panel for you to define.

6. In the **Description** box, enter the description of the action you are creating. If you entered a description when you created the action, it appears here; however, you can edit it.

Description is an optional field.

7. If you need to change the function you selected when creating the action, click  to open the **Select Function** dialog box. It lists all the functions you have created and stored in the Library.

8. In the **Caption** box, enter the name of the action as you want it to appear on the **Live** menu in LiveEditor or as you want it to appear in the pop-up tooltip, if the action appears in a custom toolbar.

Tip: Use the integrated help features in LiveEditor to find information on how actions are used in LiveEditor.

9. In the **Tooltip** box, enter additional helpful information that appears in the status bar at the bottom of LiveEditor or in the pop-up tooltip, if the action appears in a custom toolbar.

10. In the **Enable** box, select when the action is enabled for use: always or when a specific Boolean variable is true. **Always** is the default. This means the action is used whenever it is placed in DLF output. You can also click  to select a variable. When this Boolean variable is true, the action is enabled for use. The variables available depend on what you have defined in your database.

11. From the **Availability** drop-down list, select an option to limit which levels of LiveEditor, and therefore which end users, can use the action. The levels are determined by your license certificate. Select from the following options:

- **Viewer**—Viewer, Standard, and Enterprise
 - **Standard**—Standard and Enterprise. This is the default.
 - **Enterprise**—Enterprise only
12. In the **Icon to display when used in a toolbar** area, select a new icon to display when the action is used in a custom toolbar, or you can use the default icon.
For more information about adding an icon to a toolbar, see “[Specifying an Icon to Display in a Toolbar](#)” below.
13. From the Menu bar, select **File > Save**.

Note: After you have defined an action, you can call it using the **LiveAction** function.

For more information about the **LiveAction** function, see “[LiveAction](#)” on page 382.

5.3.2 Specifying an Icon to Display in a Toolbar

You can specify a custom icon that is displayed for the action when it is included in a custom toolbar. A file used for a custom icon must be no larger than 1MB. For best results, use a 16 x 16 BMP file or an ICO file that includes a 16 x 16 image. Larger images will be scaled.

The icon that you associate with an action in a toolbar also appears if you add the same action to the Live menu.

For more information about adding actions to the Live menu, see “[Adding Actions to Appear on the Live Menu](#)” on the next page.

Note: ICO files that include a 256 x 256 image in compressed PNG format are not supported.

To specify an icon to display in a toolbar:

1. In Design Manager, from the Library, drag the action to the Property Panel.
The action opens in the Property Panel for you to define.
2. In the **Icon to display when used in a toolbar (click to change)** section, click the icon button and then select an icon file.
To use the default icon after selecting a custom icon, click **Use default icon**.
3. From the Menu bar, select **File > Save**.

For more information about creating a custom toolbar, see “[Adding Actions to Appear on the Live Menu](#)” on the next page.

5.3.3 Adding Actions to Appear on the Live Menu

To add actions to the **Live** menu, you must create a new view or edit an existing view. You might add an action to the **Live** menu to give end users quick access to common tasks, such as sending an email to your support team. This topic assumes you have already created a view.

For more information about creating views, see “[Using Views to Customize the End-User Interface](#)” on page 226.

To add actions to appear on the **Live** menu:

1. In Design Manager, from the Library, drag the view to the Property Panel.
2. Add actions by dragging them from the Library (under the **Environment > Interactive > Actions** heading) to the **Custom commands** box, or by clicking .

The actions you add to the box appear on the **Live** menu in LiveEditor.

3. Use  to remove actions. Reorder actions using the  and  buttons. The actions appear on the **Live** menu in LiveEditor in the order they appear in the **Custom commands** box. You can enter a maximum of 50 actions in the **Custom commands** box.
4. From the Menu bar, select **File > Save**.

5.4 Executing Functions During LiveEditor Events

In addition to adding custom actions to Live documents, you can add custom or built-in functions that are executed automatically whenever an end user performs a specified action. For example, you can create a function that verifies that all the fields are completed before a document is saved. When you set up functions that are executed during events, you choose from a pre-defined list of supported events, such as the time when the Live document is opened or saved.

Similarly, you can set up functions to be executed when an end user makes a selection that causes an object or text area to become active. This type of execution is different from execution tied to an event, because the function is associated with a specific object, rather than an event.

For information about executing functions for specific objects, see “[Executing Functions During Editing](#)” on page 134.

To execute a function during a LiveEditor event:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
 2. Click the **Editor Framework** tab.
 3. In the **Functions to execute on LiveEditor events** section, click .
- The **Edit Settings System Event** dialog box opens. Here, you select the function and when it is executed.
4. From the **System event** drop-down list, select an option to specify the system event that will trigger the function:
 - **When document closed**—Select this option to trigger the function any time an end user closes the Live document. The function is executed regardless of the method used to close the Live document (for example, using **File > Close** or a button on the Live document).
 - **When document opened**—Select this option to trigger the function any time an end user opens the Live document.
 - **When document saved**—Select this option to trigger the function any time an end user saves the Live document.
 - **Initial document close**—Select this option to trigger the function the first time an end user closes the Live document. For example, if you want content to be sent to a repository only the first time a document is closed, select **Initial document close**. The function is executed regardless of the method used to close the Live document (for example, using **File > Close** or a button on the Live document).
 - **Initial document open**—Select this option to trigger the function when an end user opens the Live document for the first time.
 - **Initial document save**—Select this option to trigger the function when an end user saves the Live document for the first time.
 - **Customer start**—Select this option to trigger the function at the beginning of new customer data.
 - **Customer end**—Select this option to trigger the function at the end of current customer data.
 - **When data read**—Select this option to trigger the function when an end user reads a data file.
 - **When data written**—Select this option to trigger the function when an end user writes to a data file.
 - **On ODBC record update**—Select this option to trigger the function when an end user updates an ODBC database record.
 - **On ODBC record delete**—Select this option to trigger the function when an end user deletes an ODBC database record.

- **When stored**—Select this option to trigger the function when an end user stores the Live document. If you select this option, the  button is added to the Live Actions toolbar in LiveEditor.
 - **When submitted**—Select this option to trigger the function when an end user submits the Live document. If you select this option, the  button is added to the Live Actions toolbar.
 - **When printed**—Select this option to trigger the function when an end user prints the Live document.
5. In the **Associated function** field, click  to specify the function you want to be triggered at the time of the event selected from the **System event** drop-down list.
- The function must be a Boolean function that is run before the event. If it returns true, the event is allowed to occur. If it returns false, the event is cancelled. For example, if you create a custom Boolean function that calls the `LiveExternalDocImport` built-in function to import an external PDF into a placeholder document, you can specify that PDF pages will be imported into the placeholder document when an end user submits the Live document for processing. In that scenario, the end user clicking the submit button would return a value of true, and the PDF import would proceed.
6. From the Menu bar, select **File > Save**.

For more information about functions, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

5.5 Executing Functions During Editing

You can set up functions that are executed automatically when an end user makes a selection that causes an object or text area to become active, or are executed immediately after an end user makes a selection on an object. For example, suppose you want to remind end users that they should not select certain dates from a calendar since those dates are company holidays. In this case, you can set up a function that is executed when an end user opens a calendar object. The function opens a window reminding users not select the dates. When you set up functions that are executed during editing, you associate a function with an editable object. Some objects can be set up with functions that are executed when the object becomes active; on other objects, functions can be executed only when the action is completed.

Similarly, you can set up functions to be executed when a specified event occurs. This type of execution is different from execution tied to an object, because the function is associated with a specific event, rather than with a specific object.

For information about executing functions during events, see “[Executing Functions During LiveEditor Events](#)” on page 132.

To execute functions during editing:

1. In Designer, select the object or text area with which you want to associate a function.
2. Click  .
The **<Object> Properties** dialog box opens.
3. Click the **Interactive** tab.
4. Click  .
The **Advanced properties** dialog box opens.
5. In the **Start button action** box or the **Completed button action** box, click  .
The **Select Function** dialog box opens.
6. Select the function that you want to be executed when the object is activated or a selection is made on it.
7. Click **OK**.
The **Select Function** dialog box closes.
8. Click **OK**.
The **Advanced properties** dialog box closes.
9. Click **OK**.
The **<Object> Properties** dialog box closes.
10. From the Menu bar, select **File > Save**.

5.6 Using Show/Hide to Simplify Editing

You can use show and hide functionality to simplify the editing process in LiveEditor. For example, when an end user presses the DELETE or BACKSPACE key while an object or text area is selected in LiveEditor, the content is not removed; it is hidden so it is not viewable in the open Live document. You might choose to hide content if it will be included only when the end user makes a selection. In that scenario, if the end user includes content using a selection group, you can hide the content so it is visible only after the end user selects it for inclusion. Hidden content is still available if the end user opens another version of the Live document.

Note: If an object is removed because of a rule, it is considered deleted.

This section discusses the following topics:

- “[Controlling Whether End Users Can Show and Hide a Specific Object](#)” below
- “[Controlling How Hidden Objects Appear on the Outline Viewer](#)” below

5.6.1 Controlling Whether End Users Can Show and Hide a Specific Object

1. In Design Manager, from the Library, drag the object to the Property Panel.
2. Click the **Interactive** tab.
3. From the **Show and hide** drop-down list, select one of the following options to specify if and how an end user can hide an object or text area:
 - **Cannot be hidden**—Select **Cannot be hidden** to prevent end users from hiding the object or text area. This is the default setting.
 - **User can show and hide**—Select **User can show and hide** to let end users choose to show or hide the object or text area.

When you select **User can show and hide**, the **Initially hidden** check box becomes available. Select the **Initially hidden** check box if you want the object or text area to be hidden by default. On the Live Actions toolbar, end users must click  to show the hidden areas.

Tip: To let end users delete embedded objects, you must select **User can show and hide** from the drop-down list.

- **Selection group - user may pick**
 - **Selection group - system only**
4. From the Menu bar, select **File > Save**.

For more information about showing or hiding selection groups, see “[Defining the Content Selection Group Settings](#)” on page 103.

5.6.2 Controlling How Hidden Objects Appear on the Outline Viewer

You can control whether hidden objects in a document appear in the Outline Viewer when an end user first opens a Live document. This functionality controls the appearance of objects in the

Outline Viewer only. To control the appearance of objects in a Live document, use the **Show and hide** area on editable objects.

For more information about showing and hiding objects in a Live document, see “[Controlling Whether End Users Can Show and Hide a Specific Object](#)” on the previous page.

To control how hidden objects appear on the Outline Viewer in LiveEditor:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Initial behavior of hidden items** drop-down list, select **Show or Hide**.
4. From the Menu bar, select **File > Save**.

5.7 Using External Hyperlinks to Link to Locations Outside of a Live Document

Note: This section explains only external hyperlinks. For information about adding links to content that is located within the same document, webpage, or email message that the customer is currently viewing, see “[Using Internal Hyperlinks to Link to Locations Within a Live Document](#)” on page 149.

You can add external hyperlinks to text, images, shapes, and other objects in Live documents. External hyperlinks can also be included in PDF and other fulfillment outputs from a Live document. External hyperlinks can be static (so that each customer views the same link), or dynamic (so that each customer views a different link). You might use external hyperlinks to direct end users to a website that contains more information about a particular subject. For example, suppose that you work for an insurance company and you are designing a customer document that agents will send to customers as a PDF. You might want to make dozens of pages of legal information available to customers, but you do not want to include all of those pages as part of the PDF. In that case, you can add an external hyperlink to text that directs customers to the section of your company website that contains all of the legal information. When customers click the external hyperlink, they will be directed to the legal information.

In Designer, external hyperlinks can be added to text and objects that appear on a page. In Design Manager, external hyperlinks can be added to paragraph objects, text message objects, or graphic message objects, and these objects are added to documents as part of campaigns that target groups of content to specific customers. For example, in a telecommunications bill statement that is delivered to customers as a PDF, you can include promotional offers based on the current service level of each customer. The offers can include a external hyperlink to a web address that provides additional information.

For applications that produce Live documents, you can add both an external hyperlink and a Live action link to text and objects. Live action links allow end users to initiate actions, such as

calling a function or following an external hyperlink to a web address, with a single click. For example, suppose that you are designing a Live document that insurance agents will use to interview customers and explore policies and options for auto insurance. On the interview document, you could add an external hyperlink to an image of a customer's automobile. When agents select the external hyperlink, they are directed to a document that includes policy options and pricing. Agents can also select a Live action link on the same image in order to add a copy of the exact same document to the set of documents that will be assembled for the customer during fulfillment.

This section discusses the following topics:

- ["Where to Find Information About Adding External Hyperlinks to Objects in Your Design" below](#)
- ["Controlling an End Users Interaction with External Hyperlinks in Live Documents" on the next page](#)
- ["Adding Live Action Links to Text or Objects in a Live Document Using Designer" on page 140](#)
- ["Adding Live Action Links to Paragraph Objects in a Live Document Using Design Manager" on page 147](#)

5.7.1 Where to Find Information About Adding External Hyperlinks to Objects in Your Design

For additional information about adding external hyperlinks to objects in your design, see the following locations in the documentation:

Information	Overview	Where to find information
Using Designer to add external hyperlinks to objects in your design	External hyperlinks can be added to text, text boxes, static or dynamic images, polygons, predefined shapes, or charts using Designer. You can also add external hyperlinks to a text box or chart and selected text within a text box or chart.	<i>Designing Customer Communications</i> in the Exstream Design and Production documentation
Using Design Manager to add external hyperlinks to objects in your design	In Design Manager, external hyperlinks can be added to paragraph objects, text message objects, or graphic message objects, and these objects are added to documents as part of campaigns that target groups of content to specific customers.	<i>Designing Customer Communications</i> in the Exstream Design and Production documentation
Including external hyperlinks in fulfillment output from a Live document	External hyperlinks can be included in fulfillment output from a Live document to DOCX, EDGAR HTML, HTML, PDF variants (PDF, PDF/A, PDF/VT, VDX), Multi-Channel XML, and XML (composed) output types.	"Controlling an End Users Interaction with External Hyperlinks in Live Documents" on the next page

5.7.2 Controlling an End User's Interaction with External Hyperlinks in Live Documents

When you add external hyperlinks, such as www.opentext.com, to selected text within a text box in a Live document, you can choose whether an end user will follow the external hyperlink with a single click or by using **CTRL + Click**.

To control how an end user interacts with external hyperlinks:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Select the **Editor Framework** tab.
3. Choose the action for the end user to follow external hyperlinks.

To	Do this
Use CTRL + Click to follow an external hyperlink	Select the Use CTRL+Click to open hyperlinks on text check box.
Use a single click to follow an external hyperlink	Clear the Use CTRL+Click to open hyperlinks on text check box.

4. From the Menu bar, select **Edit > Save**.

For more information about adding external hyperlinks to text, images, shapes, and other objects, see *Designing Customer Communications* in the Exstream Design and Production documentation.

When you add external hyperlinks to text or objects in a Live document, keep in mind that depending on the fulfillment output type, an end user will use different actions to follow an external hyperlink address. The following table includes only general guidelines for following external hyperlinks. For output-specific information about following external hyperlinks, refer to the documentation for the software that an end user will use to view the output.

To follow an external hyperlink address, an end user must complete one of the following tasks:

To	Do this
Follow an external hyperlink address in EDGAR HTML, HTML, PDF, or VDX output	Hover the pointer over an external hyperlink and use a single left-click.
Follow an external hyperlink on text or other objects within a text box or table in DOCX output	Hover the pointer over an external hyperlink and use CTRL + Click .
Follow an external hyperlink on objects that are placed directly on a page (not within a text box or table) in DOCX output	Hover the pointer over an object with an external hyperlink, right-click, and select Open Hyperlink .

To	Do this
Follow an external hyperlink in PDF/A, Multi-Channel XML, or XML (composed) output	Keep in mind that an end user cannot follow external hyperlinks in PDF/A, Multi-Channel XML, or XML (composed) output. However, the external hyperlink address is visible when an end user hovers the pointer over the text, image, or shape that includes the external hyperlink.

5.7.3 Adding Live Action Links to Text or Objects in a Live Document Using Designer

Live action links on design objects allow end users to initiate actions, such as following an external hyperlink to go to a website (such as www.opentext.com) or calling a function, with a single click. Live action links that use external hyperlinks can be static (so that each customer views the same link), or dynamic (so that each customer views a different link). You can choose to add both an external hyperlink and a function to the same text or object. Keep in mind that Live action links can only be accessed when viewing a Live document using LiveEditor.

Note: Live action links do not support using internal hyperlinks to link within the document, webpage, or email message that the customer is viewing. Live action links support only external hyperlinks.

For more information about external hyperlinks, see “[Using External Hyperlinks to Link to Locations Outside of a Live Document](#)” on page 137.

To add Live action links to text and objects in a Live document:

1. In Designer, open any page that contains the text or object to which you want to add a Live action link.

2. To add a Live action link, complete the following tasks as needed:

To	Do this
Add a Live action link to a chart	<ol style="list-style-type: none">a. Select the chart.b. Right-click the selected chart and select Chart Properties. The Chart Properties dialog box opens.c. Click the Interactive tab.d. Click . The Advanced properties dialog box opens.e. In the Name box, specify a unique name for the chart that makes it easier to identify. This property is most often used when the chart is part of a selection group. The name that you specify might also be used by navigation functions or for troubleshooting purposes.f. If the Live action link is an external hyperlink, from the Link drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable). <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"><p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p></div> <ol style="list-style-type: none">g. If the Live action link initiates a function, use the Button action drop-down list to specify the function that runs when the interactive area is made active.h. Click OK. The Advanced properties dialog box closes.i. Click OK. The Chart properties dialog box closes.

To	Do this
Add a Live action link to an image	<ol style="list-style-type: none">a. Select the image.b. Right-click the selected image and select Image Properties. The Image Properties dialog box opens.c. Click the Interactive tab.d. Click . The Advanced properties dialog box opens.e. In the Name box, specify a unique name for the image that makes it easier to identify. This property is most often used when the image is part of a selection group. The name that you specify might also be used by navigation functions or for troubleshooting purposes.f. If the Live action link is an external hyperlink, from the Link drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable). <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"><p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p></div> <ol style="list-style-type: none">g. If the Live action link initiates a function, use the Button action drop-down list to specify the function that runs when the interactive area is made active.h. Click OK. The Advanced properties dialog box closes.i. Click OK. The Image Properties dialog box closes.

To	Do this
Add a Live action link to a polygon shape	<ol style="list-style-type: none">a. Select the polygon shape.b. Right-click the selected polygon shape and select Polygon Shape Properties. The Shape Properties dialog box opens.c. Click the Interactive tab.d. Click . The Advanced properties dialog box opens.e. In the Name box, specify a unique name for the polygon shape that makes it easier to identify. This property is most often used when the polygon shape is part of a selection group. The name that you specify might also be used by navigation functions or for troubleshooting purposes.f. If the Live action link is an external hyperlink, from the Link drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable). <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"><p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p></div> <ol style="list-style-type: none">g. If the Live action link initiates a function, use the Button action drop-down list to specify the function that runs when the interactive area is made active.h. Click OK. The Advanced properties dialog box closes.i. Click OK. The Shape Properties dialog box closes.

To	Do this
Add a Live action link to a text box	<ol style="list-style-type: none">a. Select the text box.b. Right-click the selected text box and select Text Properties. The Text Properties dialog box opens.c. Click the Interactive tab.d. Click . The Advanced properties dialog box opens.e. In the Name box, specify a unique name for the text box that makes it easier to identify. This property is most often used when the text box is part of a selection group. The name that you specify might also be used by navigation functions or for troubleshooting purposes.f. If the Live action link is an external hyperlink, from the Link drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable). <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"><p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p></div> <ol style="list-style-type: none">g. If the Live action link initiates a function, use the Completed button action drop-down list to specify the function that runs when the interactive area is made active.h. Click OK. The Advanced properties dialog box closes.i. Click OK. The Text Properties dialog box closes.

To	Do this
Add a Live action link to an interactive area	<ol style="list-style-type: none">a. Select the interactive area.b. Right-click the interactive area and select Interactive area properties. The Interactive area Properties dialog box opens.  c. Click <p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p>

To	Do this
Add a Live action link to a paragraph within an interactive area	<ol style="list-style-type: none">a. Highlight the paragraph.b. Right-click the selected paragraph and select Text paragraph properties. The Text paragraph properties dialog box opens.c. Click the Interactive tab.d. Click . The Advanced properties dialog box opens.e. In the Name box, specify a unique name for the paragraph within an interactive area that makes it easier to identify. This property is most often used when the paragraph within an interactive area is part of a selection group. The name that you specify might also be used by navigation functions or for troubleshooting purposes.f. If the Live action link is an external hyperlink, from the Link drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable). <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"><p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p></div> <ol style="list-style-type: none">g. If the Live action link initiates a function, use the Completed button action drop-down list to specify the function that runs when the interactive area is made active.h. Click OK. The Advanced properties dialog box closes.i. Click OK. The Text paragraph properties dialog box closes.

To	Do this
Add a Live action link to selected text within an interactive area	<p>a. Highlight the text.</p> <p>b. Right-click the selected text and select Text Properties. The Text Properties dialog box opens.</p> <p>c. Click the Interactive tab.</p> <p>d. Click . The Advanced properties dialog box opens.</p> <p>e. In the Name box, specify a unique name for the selected text within an interactive area that makes it easier to identify. This property is most often used when the selected text within an interactive area is part of a selection group. The name that you specify might also be used by navigation functions or for troubleshooting purposes.</p> <p>f. If the Live action link is an external hyperlink, from the Link drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable).</p> <div style="border: 1px solid black; padding: 5px;"> <p>Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for Text link and Object link in your theme are used to indicate to end users that the object is a link.</p> </div> <p>g. If the Live action link initiates a function, use the Completed button action drop-down list to specify the function that runs when the interactive area is made active.</p> <p>h. Click OK. The Advanced properties dialog box closes.</p> <p>i. Click OK. The Text Properties dialog box closes.</p>

- From the Menu bar, select **File > Save**.

5.7.4 Adding Live Action Links to Paragraph Objects in a Live Document Using Design Manager

You can use Design Manager to add a Live action link to paragraph objects in a Live document. Live action links on paragraph objects initiate actions such as following an external hyperlink to go to a website (such as www.opentext.com). Live action links that use external hyperlinks can be static (so that each customer views the same link) or dynamic (so that each customer views a different link).

Note: Live action links do not support using internal hyperlinks to link within the document, webpage, or email message that the customer is viewing. Live action links support only external hyperlinks.

For more information about external hyperlinks, see “[Using External Hyperlinks to Link to Locations Outside of a Live Document](#)” on page 137.

To add a Live action link to a paragraph object in a Live document:

1. In Design Manager, from the Library, drag the object to the Property Panel.
2. Click the **Interactive** tab.
3. Click  .
The **Advanced properties** dialog box opens.
4. In the **Name** box, specify a unique name for the object or text area that makes it easier to identify. This property is most often used when the object or text area, usually a paragraph in a text box, is part of a selection group. The name you specify might also be used by navigation functions or for troubleshooting purposes.
5. From the **Link** drop-down list, select whether the external hyperlink address is static or dynamic (controlled by a variable).

Note: Depending on what you select, the text box or the variable selection box becomes active. The visual indicators that you specified for **Text link** and **Object link** in your theme are used to indicate to end users that the object is a link.

6. Click **OK**.
The **Advanced properties** dialog box closes.
7. From the Menu bar, select **Edit > Save**.

For more information about themes, see “[Using Themes to Visually Guide End Users](#)” on page 218.

Note: You can also use the `LiveLaunchURL` function to launch destinations automatically.

For more information about the `LiveLaunchURL` function, see “[Live Built-In Functions](#)” on page 381.

5.8 Using Internal Hyperlinks to Link to Locations Within a Live Document

Note: This section explains only internal hyperlinks. For information about adding links to content that is located outside of the document, webpage, or email message that the customer is viewing, see “[Using External Hyperlinks to Link to Locations Outside of a Live Document](#)” on page 137.

You use internal hyperlinks to direct customers to content that is located within the document, webpage, or email message that they are currently viewing. For example, suppose that you are creating a stock portfolio summary document that must cite specific sections, images, or charts in order to highlight important information for the customer to reference, but that information is spread across multiple pages; you could use internal hyperlinks to redirect customers to those specific locations within the content.

Internal hyperlinks are visible in the document content only during design and fulfillment. This means that while you can view and edit internal hyperlinks while you are designing the Live document, end users cannot view or interact with internal hyperlinks in LiveEditor. However, after the content is saved as PDF output or after the content has been fulfilled to PDF, HTML, or HTML (email) output, the end user can view and interact with the internal hyperlink. This unique behavior means that as a designer, you must keep in mind some special considerations about the end user experience when creating text that contains internal hyperlinks.

This section discusses the following topics:

- “[Where to Find Information About Adding Internal Hyperlinks to Objects in Your Design](#)” below
- “[Best Practices For Including Internal Hyperlinks in Live Documents](#)” on the next page

5.8.1 Where to Find Information About Adding Internal Hyperlinks to Objects in Your Design

For additional information about adding internal hyperlinks to objects in your design, see the following locations in the documentation:

Information	Overview	Where to find information
Adding internal hyperlinks to text and objects	You can add internal hyperlinks to text and objects in Designer.	<i>Designing Customer Communications</i> in the Exstream Design and Production documentation

Information	Overview	Where to find information
Adding hyperlink anchors to text	You can apply hyperlink anchors to text to mark the destination of the internal hyperlink. You cannot apply internal hyperlinks to objects such as images, charts, or shapes. However, if you want to direct a customer to a design object, you can achieve this by applying a hyperlink anchor object to text placed near the design object.	<i>Designing Customer Communications</i> in the Exstream Design and Production documentation
Formatting the appearance of internal hyperlinks	You have the option to format the appearance of internal hyperlinks by customizing the appearance of the text.	<i>Designing Customer Communications</i> in the Exstream Design and Production documentation
Including internal hyperlinks in output	Internal hyperlinks are supported in PDF, HTML, and HTML (email) output only.	<i>Designing Customer Communications</i> in the Exstream Design and Production documentation

5.8.2 Best Practices For Including Internal Hyperlinks in Live Documents

While you can create designs that contain internal hyperlinks, end users cannot view or edit internal hyperlinks while working in LiveEditor. As a best practice, you should not place internal hyperlinks or anchors in editable areas. In LiveEditor, internal hyperlinks appear with no visual indicator to show that text or objects are affected by an internal hyperlink or anchor. If end users edit text or objects that contain an internal hyperlink or anchor, they can unintentionally cause unexpected results in the final output, with no indication that they might be causing an issue.

End users can, however, view internal hyperlinks in the final output of the Live document when they choose to save as PDF, or if they generate PDF, HTML, or HTML (email) output during fulfillment. This can be a useful if you want to view how changes made in LiveEditor (such as editing content or adding pages) will affect internal hyperlinks in the final output.

5.9 Making Data Available in Live Documents

Depending on your design and data needs, you can provide a number of ways for end users to access the data needed to complete a Live document. For example, if end users will access a Live document outside of your company's firewall, you can embed data files within a DLF so that the necessary data will be available from within LiveEditor. In addition, you can allow end users to add their own data to a Live document by giving them the ability to read data files or upload distribution lists.

This section discusses the following topics:

- “[Embedding Data Files](#)” below
- “[Adding Authorized Data Files to a Live Application to Allow End Users to Access Data](#)” on page 153
- “[Allowing End Users to Upload Distribution Lists](#)” on page 155

Note: You can also use the `LiveDataFileClose` and the `LiveDataFileOpen` built-in functions to open and close files (for example, by using an edit action).

For information about the `LiveDataFileClose` and the `LiveDataFileOpen` built-in functions, see “[Live Built-In Functions](#)” on page 381.

5.9.1 Embedding Data Files

You can make DLF files more portable by embedding data files (specifically, reference and initialization files) within them. As a result, it is not necessary for end users in LiveEditor to be logged in to a company's network in order to access the data available in reference and initialization files. Instead of specifying a network path to a data file in Design Manager, you can embed the data file and allow users to access the data from within LiveEditor.

To embed a reference or initialization file within a DLF file:

1. In Design Manager, from the Library, drag the reference or initialization file object to the Property Panel.
2. Click the **Basic** tab.
3. From the **Access** drop-down list, select **Memory**.

For more information about how the engine stores and accesses data from a reference file, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

4. Click the **Interactive** tab.
5. Select the **Place a copy of the data file in the Live Document** check box.

The other options on the **Interactive** tab become available.

6. If necessary, select the **Use the embedded data in fulfillment** check box.

If you select this option, the embedded data file is used for any applications that run after the DLF is submitted for fulfillment. You might select this option if the fulfillment application adds additional information (such as a page or a document) based on the selections that end users make in LiveEditor.

You can also use the `DISABLE_EMBED_DATA` engine switch to disable the use of an embedded reference file in a fulfillment application.

For more information about the DISABLE_EMBED_DATA engine switch, see “[DISABLE_EMBED_DATA](#)” on page 439.

Note: Because embedded initialization files are used only during the initial engine run, they cannot be used for fulfillment.

7. From the **IO times** section, select one or more options to specify the time at which the embedded data will be read within LiveEditor:
 - **Document open**—Select this option to read the embedded data file each time that the Live document is opened.
 - **Initial open**—Select this option to read the embedded data file only the first time the Live document is opened.
 - **Data file menu**—Select this option to read the embedded data file only when end users select the data file from the LiveEditor Menu bar (by selecting **Live > Data file > Read > [data file name]**).
 - **When key changes**—Select this option to read the embedded data file only when the key variable changes in a reference file.
8. From the Menu bar, select **File > Save**.

When working with embedded data files, keep in mind the following behaviors:

- If a data file is embedded from the **Interactive** tab and it is included as an authorized data file on the Live setting object, then the embedded file settings will be honored and the authorized data file settings will be ignored.

For information about adding authorized data files, see “[Adding Authorized Data Files to a Live Application to Allow End Users to Access Data](#)” on the next page.

- During fulfillment, an embedded reference file applies only to customers from its parent driver file.
- If the object OI for the embedded reference file matches the object OI for the reference file from the fulfillment application, then the following conditions apply:
 - If the **Place a copy of the data file in the Live Document** or the **Use the embedded data in fulfillment** check boxes are not selected, then the embedded file is ignored.
 - If the **Place a copy of the data file in the Live Document** and the **Use the embedded data in fulfillment** check boxes are selected, then the mapping and data for the embedded file replaces the mapping and data for the reference file in the fulfillment application.

(To find the object OI, right-click on the reference file and select **Administer**. The number in the **Internal ID** field is the OI for the object.)

5.9.2 Adding Authorized Data Files to a Live Application to Allow End Users to Access Data

You can add authorized data files to a Live application so that end users can access additional data not contained in the DLF file. These files can be accessed automatically based on end user actions, or you can make them available for end user selection. Remember, however, that any authorized data files that you add must use the same layout as the data files used in the DLF file. For example, suppose you use the same letter for both corporate and residential customers, but the two sets of customers have different data files. Both data files have the name and address fields in the same order, so you can allow end users to read the correct data file as necessary.

To add authorized data files to a Live application:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Data** tab.
3. In the **Authorized data files** section, click .

The **Edit Settings File Authorization** dialog box opens to let you select which data file to add and when end users can access the data file.

4. In the **Data file** box, click  and select the data file you want to enable for end users.

The **Select Data File** dialog box opens to let you choose the data file.

5. In the **IO times** box, specify the time at which the selected data file is accessed. Select one or more of the following options:
 - **Document open**—Select this option to read the embedded data file each time that the Live document is opened.
 - **Document close**—Select this option to read the embedded data file each time that the Live document is closed.
 - **Initial open**—Select this option to read the embedded data file only the first time the Live document is opened.
 - **Initial close**—Select this option to read the embedded data file only the first time the Live document is closed.
 - **Data file menu**—Select this option to read the embedded data file only when end users select the data file from the LiveEditor Menu bar (by selecting **Live > Data file > Read > [data file name]**).
 - **When key changes**—Select this option to read the embedded data file only when the key variable changes in a reference file.

Caution: The **When key changes** option is not supported on reference files that use multiple variables for keys.

The check boxes you select in the **IO times** box specify the time at which the selected data file is accessed. However, the type of data file you select determines the options available. The following table lists the IO times (marked with an X) that are available for each type of data file.

IO times available per data file type

Data file type	Document open	Document close	Initial open	Initial close	Data file menu	When key changes
Customer driver file	X		X		X	
Initialization file	X		X		X	
Post-sort initialization file	X		X		X	
Post-sort report file	X	X	X	X	X	
Reference file	X		X		X	X
Report file	X	X	X	X	X	

For more information about data file types, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

6. In the **Path** box, enter the starting path that appears in the **Open File** dialog box in LiveEditor. Enter the path that is most likely to have the data file the end user is reading.
7. If you selected the **Data file menu** check box in step 5, additional options become available. Enter values for the following options as necessary:
 - In the **Caption** box, enter the name of the data file as you want it to appear on the **Live > Data file > Read** menu in LiveEditor.
 - In the **Tooltip** box, enter additional helpful information that you want to appear in the status bar at the bottom of LiveEditor.
 - In the **Enable** box, select when the data file is enabled for use: always or when a specific Boolean variable is true. **Always** is the default. **Always** means the data file is used whenever it is placed in DLF output. To select a variable, you can also click . When this Boolean variable is true, the data file is enabled for use. The variables that are available depend on what you have defined in your database.
8. From the Menu bar, select **File > Save**.

Specifying the File Type of User-Added Data Files

If you want to allow end users to add their own data files to the Live document, you can use the **LiveEditor parameters** area on the **Interactive** tab of the data file properties to describe the type and file extension of the file used to create the data file. This option allows end users to select other files of the same type to populate the data file object.

To specify the file type of user-added data files:

1. In Design Manager, from the Library, drag the data file object to the Property Panel.
2. Click the **Interactive** tab.
3. In the **LiveEditor parameters** area, specify description and extension of the file used to create the data file.
4. From the Menu bar, select **File > Save**.

5.9.3 Allowing End Users to Upload Distribution Lists

If you are creating a Live document that is a template and allows end users to add data files, you can use the properties on the Live setting object to let end users add data using a distribution list. For example, suppose you want to create a template for a promotional flier for your branch offices around the world, but each office maintains its own list of customers. In this case, each office would update the flier and upload its own distribution list.

For information about enabling distribution lists, see “[Processing Live Documents](#)” on page 360.

Chapter 6: Allowing End Users to Import/Export Components of a Live Document

One of the advantages of the DLF file structure and the editing process enabled by Live documents is that the document is "open" and can be changed throughout its life cycle by many users. These changes can be small, such as editing wording or inputting data, or they can be large, such as adding entire paragraphs or even pages of content. Exstream Live provides several ways you can allow end users to import content during the editing process to help save time and prevent errors from occurring during the editing. For example, if your organization stores snippets of pre-composed wording in content repository, you can set up a web service that will allow end users to pull a specified snippet into a Live document.

In addition to allowing end users to import content, you can also set up the Live document to allow end users to export content to specific file formats. For example, you might want to allow end users to export a snippet of content they have edited to be stored in a content repository so it can be used in other documents.

The following table lists the ways you can allow end users to import or export components of a Live document and when each method might best be used.

Import/Export methods

Type of import/export	Description
Importing unformatted text	<p>This method is the most basic way to allow end users to import content as they edit a Live document. Using this method, end users can place the cursor in an editable text box and then choose a locally stored text file from which to import content. After the text is imported, end users can then apply formatting so the content is consistent with the appearance of the Live document.</p> <p>You might use this method if you want to give end users the freedom to make extensive changes to text in a Live document. By allowing them to import content from a text file, they can author text in other programs and reuse it in multiple Live documents, without having to manually enter it.</p>
Importing formatted text (that will not be edited)	<p>Just as you can in traditional Exstream Design and Production designs, you can use tagged text variables to import formatted text into Live documents (for example, using a web service to import snippets of tagged text from a repository). This method allows you to leverage content you have stored in tagged text format for use in Live documents. Keep in mind, however, that this method is not suitable for content that will be edited in LiveEditor; if you want to import and edit formatted text in LiveEditor, you must use a formatted text variable.</p> <p>For more information about using tagged text variables, see <i>System Administration</i> in the Exstream Design and Production documentation.</p>

Import/Export methods, continued

Type of import/export	Description
Importing or exporting formatted text (that can be edited)	<p>This method allows you to set up areas in which end users can import or export formatted text. To enable this functionality, you use a special type of variable called a formatted text variable. This variable stores the content that is imported or exported so it can be accessed later in the document process.</p> <p>You might use this method if you store snippets of pre-composed wording in a content repository. By allowing end users to import this formatted content into a Live document, you can ensure that all users are using the same content in Live documents. This method also helps you prevent errors that can occur during the editing process, and can help ensure that corporate formatting policies are enforced.</p>
Importing pages	<p>Similar to the dynamic content import feature of Exstream Design and Production, this method allows you to set up areas in which end users can import entire pages into a Live document. You use placeholder documents to designate areas within the Live document where end users can import external pages.</p> <p>You might use this method if you create large documents, such as contracts or policies, that contain not only boilerplate information that applies to all customers and products, but also information specific to each customer. This method can enable end users to import pages containing boilerplate information, while still giving them flexibility to edit and customize other areas of the Live document.</p>

This chapter discusses the following topics:

- “[Allowing End Users to Import Unformatted Text](#)” below
- “[Allowing End Users to Import and Export Formatted Text](#)” on the next page
- “[Importing Pages from External Files into a Live Document](#)” on page 166

6.1 Allowing End Users to Import Unformatted Text

1. In Designer, select the text box in which you want to allow end users to upload text.
2. Click . The **Text Properties** dialog box opens.
3. Click the **Interactive** tab.
4. From the **Content change** drop down list, select either **Change is optional** or **Change is required**, as needed.
Additional options become available.
5. Select the **Text edit** check box.
Additional options become available.
6. From the **Use control to select content** drop-down list, select **File upload**.

7. In the **Prompt to display when there is no image** box, enter text that you want to appear in the area to indicate that end users can import text.

8. Click **OK**.

The **Text Properties** dialog box closes.

9. From the Menu bar, select **Edit > Save**.

6.2 Allowing End Users to Import and Export Formatted Text

You can make it faster and easier for end users to contribute content to documents by setting up Live documents in which they can import and/or export text. Designed specifically for organizations in which individual pieces of content are stored in repositories, the importing and exporting feature of Exstream Live allows end users to leverage pre-designed content in their documents, as well as contribute new content that can be stored and then used in other document processes. For example, suppose your organization visits customer sites every quarter to gather financial data. An end user can open a Live document template into which the previous quarter's information for that customer is imported. The end user can add the new data and export the new content back to the repository where it can be stored and used for other purposes, such as an end-of-year statement.

The importing and exporting feature uses a special type of variable called a "formatted text variable," which preserves the formatting of text that is imported and exported. Formatted text variables support the importing of content using local file references or web service calls. Several file types can be imported and/or exported using a formatted text variable; these file types and the requirements for using each are discussed in a later section.

In addition to using this feature to allow end users to import and export formatted text while editing a Live document, you can also use a formatted text variable to import text during the initial engine run, just as you can with placeholder variables.

For more information about importing content during an engine run, see *Importing External Content* in the Exstream Design and Production documentation.

This section discusses the following topics:

- “[Technical Requirements for Setting Up the Ability to Import/Export Formatted Content](#)” on the next page
- “[Designing an Application to Allow Formatted Text to be Imported/Exported](#)” on page 162

6.2.1 Technical Requirements for Setting Up the Ability to Import/Export Formatted Content

Exstream Live supports importing/exporting content in the following file types:

- Base-64 encoded
- DXF (composed XML)
- Plain text
- Rich Text Format (RTF)

As you design a document that allows end users to import/export content, keep in mind the following general considerations:

- Not all file types allow you to retain all formatting. For best results, review the information on supported formatting before you set up your application to ensure that the formatting you require is supported in the file format you want to use.

For more information about the formatting retained for each file type, see ["Specifications for Individual File Types" below](#).

- You can import and export from single-byte format to double-byte format and vice versa. However, if you import double-byte content to single-byte format, non-ASCII characters are not imported correctly.
- All file types except for Base-64 encoded allow the content to be edited both in Live documents as part of the editing process, and outside of the Live environment. Base-64 encoded content can be edited only in Live documents.

You should also review the specifications for individual file types before you set up importing/exporting in your Live document.

Specifications for Individual File Types

This section discusses the requirements for using specific file types, and lists the supported formatting for each type.

- If you import/export Base-64 encoded content, keep in mind that you cannot import DBCS data into an SBCS DLF. However, if the SBCS and DBCS versions of a DLF are created from the same application, you can import SBCS data into a DBCS DLF.
- If you import/export Base-64 encoded content, keep in mind that Base-64 encoded content can be edited only in a Live document.
- If you import/export DXF content, keep in mind that the variables used to import DXF content cannot be pre-populated during the initial engine run. They can be populated only in LiveEditor.

- If you import/export DXF content, you must convert the XML to valid DXF format using XSL transformation (XSLT) or another tool. LiveEditor does not convert XML to DXF format. If the content is not in valid DXF format, you receive an error and the content will not be imported.
- If you import/export DXF content, you should familiarize yourself with the DXF specification to make sure that the design objects you want to import are supported by the DXF file format.

The following table lists the formatting features (marked with an X) that are maintained when text is imported or exported using a formatted text variable. Features with an asterisk (*) must be included in the design page.

Supported format features for formatted text variables

Feature	Base-64 encoded content	DXF	Plain text	RTF
Carriage returns/line feeds	X	X	X	X
Tabs	X	X	X	X
Indentations	X	X		X
Horizontal alignment	X	X		X
Font face (font name)*	X	X		X
Font size*	X	X		X
Font color*	X	X		X
Bold*	X	X		X
Underlining*	X	X		X
Bold and italics*	X	X		X
Bold and underlining*	X	X		X
Bold, italics, and underlining*	X	X		X
Italics and underlining*	X	X		X
Superscript	X	X		X
Subscript	X	X		X
Single-tier bullets (all bullets drawn as filled)	X	X		X
Multi-tier bullets	X	X		X

Supported format features for formatted text variables, continued

Feature	Base-64 encoded content	DXF	Plain text	RTF
Horizontal lines (drawn)	X	X		
Revisions (for black lining)	X			
Single-tier numbered lists	X	X		
Multi-tier numbered lists	X	X		
Special characters	X	X		
Embedded objects (including images and tables)		X		
Background color (highlight)		X		

Converting Content in Various Formats Using Logic

You can use rules, formulas, and functions to manipulate and convert formatted content. One way to use this functionality is to convert content stored in less flexible formats, such as tagged text, that you would not normally be able to import as editable text. You can use conversion logic to convert this content to formatted text and leverage it in Live documents. The following table lists valid text conversion combinations:

Text conversion combinations

Conversion	Syntax description
Convert to formatted text	<code>FormattedTextVar = StringVar</code>
Convert to formatted text	<code>FormattedTextVar = TaggedTextVar</code>
Convert to plain text string	<code>StringVar = FormattedTextVar</code>
Convert to tagged text	<code>TaggedTextVar = FormattedTextVar</code>
Convert formatted text variable to DXF XML string	<code>StringVar = FORMAT(FormattedTextVar, "DXF")</code>
Convert DXF XML string to formatted text variable	<code>MAPFORMAT (StringVar, LayoutWithDXFFormattedTextVar)</code>
Convert to plain text string	<code>StringVar = FORMAT(FormattedTextVar, "PLAINTEXT")</code>

Note: All other data types (Boolean, currency, date, float, integer) will be converted to and from text by way of a pre- or post-conversion to a string.

6.2.2 Designing an Application to Allow Formatted Text to be Imported/Exported

To design an application to allow formatted text to be imported/exported, you must complete the following tasks:

1. [“Setting Up a Formatted Text Variable” below](#)
2. [“Setting Up Data Files to Support Importing and Exporting” on the next page](#)
3. [“Setting Up the Importing and Exporting Method” on page 165](#)
4. [“Setting Up the DLF Output Object” on page 166](#)

In addition, when you set up the page that will contain the imported/exported content, make sure to take into consideration the content that will be added. If formatted content will be imported, you might want to use styles that will ensure the rest of the page layout will look consistent with the imported content. Also, set up any necessary Trigger functions to automate the import/export actions, if needed.

Setting Up a Formatted Text Variable

Formatted text variables provide the unique capability to allow content from specified file types to be imported into a Live document while the formatting of that content is retained. Formatted text variables also allow content to be exported from a Live document and returned to a file location or repository. Similar to placeholder variables, which are placed on pages to allow content to be imported at a particular location, formatted text variable are placed in editable text boxes, text paragraphs, or table cells on a page. When the importing of content is triggered in the LiveEditor, the specified content is imported to the location reserved by the formatted text variable. Similarly, when content is exported, the content stored by the formatted text variable is exported to the specified destination.

You create and set up formatted text variables as you would create and set up other types variable. However, keep in mind the following considerations:

- In order to create and use formatted text variables, you must have licensed the Dynamic Content Import module.
- When you create the variable, make sure you select **Formatted Text** from the **Type** drop-down list. This defines the variable as one that allows text to be imported or exported.
- If you want the variable to control the importing of multiple, consecutive objects, such as table rows or paragraphs, you must define the variable as an array by selecting the **Array** check box on the **Basic** tab of the variable properties.

- You can use formatted text variables in the same way you can use other types of variables to make sure that the text entered is valid or of the correct length. For example, if a paragraph cannot exceed 300 characters, and the end user enters 320 characters, the paragraph is invalid.
- The encoded string length of formatted text variables cannot exceed 4MB.

For more information about defining the basic properties of a variable, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

When you map the data file, the following formatting options are available to you on the **Data Area Properties** dialog box:

- **Base-64 Encoded Content**
- **RTF Content**
- **Plain Text Content**
- **Composed XML (DXF) Content**

Note: The **Format** type you select on the **Data Area Properties** dialog box and the **Special Formatting** type you select on the **Variable Properties** dialog box within Designer must be the same. For example, if you choose Base-64 encoded content as the format type in Design Manager, you must choose Base-64 encoded content as the special formatting type in Designer.

Setting Up Data Files to Support Importing and Exporting

The ways in which end users can interact with content stored in a formatted text variable are controlled by the way you set up the data files and the application. You can use different data file combinations to enable the importing and exporting of content. The following table lists the import and export methods you can use together:

Import and export methods

With this import method	Use this export method
Customer driver file	Report file (You must export each customer's content individually.)
Reference file	Report file (You must export each customer's content individually.)
Initialization file	Report file (You must export each customer's content individually.)
LiveXMLRead built-in function	LiveXMLWrite built-in function
The read content XML function (available on the Live menu in LiveEditor)	The write content XML function (available on the Live menu in LiveEditor)

Import and export methods, continued

With this import method	Use this export method
A reference file triggered from LiveEditor	<p>Reference file</p> <p>Note: The only way to update an external non-ODBC reference file is to use a connector.</p>

Note: The import methods for DXF content are limited to using initialization files and reference files. To export DXF content, you can use report files and reference files.

The following table explains ways you can set up your application to support the type of editing capabilities you require. You can use any of the valid data file combinations to enable editing; the data files used in the following table are meant as examples only.

Options for controlling how end users edit text

To	Do this
Let end users edit text that is imported and exported using the Read and Write options on the Live menu within LiveEditor	<p>Create a customer driver file using a file format of XML data file and map the contents of the file to the formatted text variable.</p> <p>The customer driver file must use a layout in which the outermost tags are <code><Customer></code>.</p> <p>For example:</p> <pre><Customer> <Data> Information </Data> </Customer></pre> <p>If the DXF content is used in an XML data file, the layout must be structured as follows:</p> <pre><Customer> <Data> <![CDATA[Information]]> </Data> </Customer></pre> <p>LiveEditor formats content this way automatically when writing to a report file.</p>
Let end users edit imported text	<ol style="list-style-type: none"> 1. Create an initialization file and map the contents to the formatted text variable. 2. Create a report file and map the contents to the formatted text variable. 3. Create two Trigger functions: one for the initialization file and one for the report file. 4. Use the Trigger functions to specify the timing of the import and export. For example, you can have the formatted text be imported when the document opens and exported when the document closes using LiveEditor events. <p>For more information about customer driver files, initialization files, and report files, see <i>Using Data to Drive an Application</i> in the Exstream Design and Production documentation.</p>

Options for controlling how end users edit text, continued

To	Do this
Let end users manually enter text	<ol style="list-style-type: none"> 1. Create an initialization file and map an empty file to the formatted text variable. 2. Create a report file and map the empty file from step 1 to the formatted text variable. 3. Create two Trigger functions: one for the initialization file and one for the report file. 4. Use the Trigger functions to specify the timing of the import and export. For example, you can have the formatted text be imported when the document opens and exported when the document closes using LiveEditor events.

Setting Up the Importing and Exporting Method

You can either use a web service or a traditional file reference to import or export formatted content, depending on your environment and workflow. For example, if you are importing formatted content during the initial engine run, you might choose to use a reference file to pull in content stored locally. However, if you want end users to import content from a enterprise repository, you might need to use a web service to access it.

Importing or Exporting Formatted DLF Content Using a Web Service

This task provides an overview of the process you must complete to set up a web service to import and export content. Before you set up the importing or exporting method, make sure the Live document is designed to support data submission. For example, include a button or menu item to allow end users to import content to the Live document.

To import or export formatted DLF content using a web service:

1. Create and define an auxiliary layout file for web services. This is the web service request.
2. Create and define a reference file for web services. This is the web service response.
3. Map the data files based on the design of the Live document:

To	Do this
Import formatted text using a web service	Map the reference file to the formatted text variable.
Export formatted text using a web service	<ol style="list-style-type: none"> a. Map the auxiliary layout file to the formatted text variable. b. On the Basic tab of the reference file properties, from the Reference key layout drop-down list, select the auxiliary layout file.

Note: You must map a request and response for each web services call. For example, if you design a Live document that imports content automatically when a Live action is initiated and exports content automatically when the Live document is saved, you must map two requests and two responses.

For more information about submitting DLF content to a web service, see “[Using a Web Service to Exchange Data Between Live Documents and Other Enterprise Systems](#)” on page 188.

Importing or Exporting Formatted DLF Content Using a File Reference

To use a file reference to import or export formatted content, define a data file as you would define any data file. Then, map the data file to the formatted text variable.

For more information about mapping data files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

Setting Up the DLF Output Object

When you create the application for importing and exporting formatted text, you create and define a DLF output object as you normally would. Keep in mind that if you want to include XML data to allow end users to read and write XML content, you must also do the following:

1. On the **Basic** tab of the DLF output properties, make a selection from the **Allowed use** drop-down list to specify which parts of the DLF (content, data, or both) can be accessed.
2. Select the **Include data** check box.
3. In the **Data file for XML layout** box, select the customer driver file you created for this application.
4. From the Menu bar, select **File > Save**.

For more information about setting up a DLF output object, see the “[Setting Up the Objects Necessary for Live Output](#)” on page 354.

6.3 Importing Pages from External Files into a Live Document

You can add placeholder documents to a Live document so that end users can import external DOCX, DXF, PDF, or TIFF files as pages within the document. The pages can be added either automatically or manually without having to regenerate the file. This can be especially useful with larger document applications, such as contracts or policies that contain not only boilerplate information that applies to all customers and products, but also information that is specific to each customer or product.

Using placeholder documents, you can set up Live document templates that contain different types of documents. For example, a Live document could contain not only designed documents that take advantage of the controlled editing environment available in LiveEditor, but also placeholder documents that allow LiveEditor users to import customer- or product-specific DOCX, DXF, PDF, or TIFF files as necessary.

When you add placeholder documents to a design, you have two options for determining how they will handle external files:

- **Automatic import**—Placeholder documents can be designed so that they import external files automatically when users perform certain tasks in LiveEditor, such as submitting a Live document for processing. For example, suppose that your insurance company provides auto, homeowners, and life insurance for customers throughout the country/region. You might need to set up a DLF template that includes policy information about your company that pertains to all customers in all areas of the country/region. If your company frequently updates this information and keeps the file in a shared location on a network drive, you could set up a placeholder document that automatically imports a file as soon as a LiveEditor user submits the Live document for processing.
- **Manual import**—In the same insurance company scenario, you could set up placeholder documents that allow agents across the country/region to supplement automatically imported pages with manually imported customer- and policy-specific pages. These Live documents would contain placeholder documents that agents could use to import external files themselves as necessary.

Note: You must have licensed the PDF Import as Image module if your Live document contains imported PDF pages and will be fulfilled for output on an AFP printer.

This section discusses the following topics:

- “[About Using Placeholder Documents to Import Pages into a Live Document](#)” below
- “[Automatically Importing External Pages into a Live Document](#)” on page 169
- “[Allowing End Users to Import External Pages into a Live Document](#)” on page 173
- “[Setting Up a Placeholder Document to Allow Imported DOCX or DXF Files to Flow](#)” on page 175

6.3.1 About Using Placeholder Documents to Import Pages into a Live Document

Like other documents, placeholder documents can contain single or multiple pages, which allow single- or multiple-page DOCX, DXF, PDF, or TIFF files to be imported into a Live document. When you use a placeholder document, you associate it with a placeholder variable, which specifies the properties of the external file that can be imported into the placeholder document in LiveEditor. For multiple-page file imports, the placeholder variable should always be specified as an array, so that every page in the file will then become one element in the array. If the placeholder variable for a multiple-page placeholder document is not specified as an array, only the first page of the imported file will be added to the document.

Placeholder documents are used elsewhere in Exstream Design and Production, but in a slightly different way: whereas placeholder documents used in other places in Exstream are fulfilled

during the initial engine run, those used in Live documents can be added during the editing session. However, the process for setting up placeholder documents is identical.

For more information about how placeholder documents are used elsewhere in Exstream, see *Importing External Content* in the Exstream Design and Production documentation.

As you implement placeholder documents into your Live documents, keep in mind the following behaviors:

- A single placeholder document can hold only one type of file. For example, if a placeholder document contains pages from a PDF file, you can only append additional pages from other PDF files to that placeholder document. If a DLF requires you to import more than one type of file, then you must create a separate placeholder document for each file type you want to import.
- Placeholder documents that contain section data behave differently than standard placeholder documents in the following ways:
 - Normally, placeholder documents whose variables do not have a value that will be populated in the engine (such as those that allow end users to import external pages manually) do not appear in the Outline Viewer panel in LiveEditor until end users add them to the DLF. However, if those same placeholder documents contain variables that will not be populated in the engine but are driven by sections, then they will appear in the object tree as placeholder documents containing a single blank page. End users can replace or append the pages of these placeholder documents in LiveEditor.
 - Placeholder documents that contain section data cannot be moved or deleted using the Outline Viewer panel in LiveEditor.

For information about using section data in a Live document, see ["Using Section Data with Live Documents" on page 200](#).

- External PDFs and TIFFs are imported into placeholder documents as image files, which means that they cannot be manipulated using LiveEditor tools. If a PDF or TIFF requires changes, users must make the changes in the program used to create the file so that the updated file can then be re-imported into the Live document.
- If the DLF file will be processed by the engine, color or grayscale TIFF files must use CCITT Group 4 compression, and black-and-white TIFF files must either be uncompressed or use CCITT Group 4 compression. Color or grayscale TIFF files that are uncompressed or use LZW compression appear correctly in LiveEditor, but they cannot be included in output during fulfillment.
- If you import a DOCX or DXF file that contains variables, the variables within the content will automatically update to reflect the variable values that are used in the DLF. For example, if the 'CustomerLanguage' variable in the DLF is set to French, then the same variable in the DXF file will also be set to French after the file is imported. No other content within the imported pages of a DOCX or DXF file can be manipulated using LiveEditor tools. If a DOCX or DXF file requires any other changes, users must make the changes in the program that is used to create the file so that the updated file can then be re-imported into the Live document.

- External files that are imported into a placeholder document can be accessed only by the Live fulfillment engine and cannot be extracted for use in other processes.
- Importing external files into a placeholder document increases the file size of the DLF. Larger file sizes can affect engine processing time, as well as the performance of the Live document in LiveEditor.
- The page numbers on individual pages and on the table of contents in a Live document are automatically updated when new pages are imported into a placeholder document.
- On a z/OS system, you cannot fulfill a Live application from a DLF file that contains imported PDF content. The z/OS engine accepts only DOCX, DXF, and TIFF files as external content for placeholder documents.

For more information about format-specific considerations for importing DOCX, DXF, PDF, or TIFF files, see *Importing External Content* in the Exstream Design and Production documentation.

6.3.2 Automatically Importing External Pages into a Live Document

When designing a template, you might want to set up a placeholder document that automatically imports a particular file into the Live document.

To use a placeholder document to import files automatically, you must complete the following tasks:

1. [“Creating a New Placeholder Variable” on page 174](#)
2. [“Creating a New Placeholder Document in Design Manager” on page 174](#)
3. [“Creating a Function to Drive Automatic Page Import” on page 171](#)
4. [“Assigning a LiveEditor Event That Will Trigger the Import Function” on page 172](#)

Creating a New Placeholder Variable

You must create a variable that defines the properties of the external file(s) that will be imported into the Live document. The placeholder must use the following settings:

- The **Type** must be **Placeholder**.
- The **Placeholder Type** must be one of the following options:
 - **DXF**
 - **PDF**
 - **TIFF (B&W G4 or uncomp)**

- **TIFF (Color)**
 - **Word (*.docx)**
- If the imported file contains multiple pages, you must select the **Array** check box.

You can define the other variable properties as needed.

Creating a New Placeholder Document in Design Manager

You must create a placeholder document that will hold the content from the external files being imported. This document will use the newly created placeholder variable to define the type of external file that it can accept.

To create a new placeholder document in Design Manager:

1. Create a new document.

For information about creating documents, see *Designing Customer Communications* in the Exstream Design and Production documentation.

2. On the **Basic** tab of the document properties, from the **Document type** drop-down list, select **Placeholder (use pre-composed content)**.
3. In the **Placeholder** box, define the variable used to import the DLF.

- a. In the **Placeholder variable** box, click .

The **Select Variable** dialog box opens.

- b. Browse to the placeholder variable you created for the external file.
- c. Click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Placeholder variable** box.

4. Define the other document properties as needed.
5. Add a page to the document to make it structurally valid. The name you assign to a page will be applied to all pages imported into that document in LiveEditor.

Tip: If you want to include additional content for the customer, add those pages to the document or to a different document as needed.

For information about adding pages and defining general document properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

6. If the document will contain multiple pages, you must configure the page object to accept multiple pages.

- a. From the Library, drag the document to the Edit Panel.

A graphical representation of the document appears.

- b. In the right column, double-click the page name.

The **Document Page Properties** dialog box opens.

- c. From the **Position of page in document** drop-down list, select **Filler Page (as Required)**.

This option allows the page to duplicate itself so that it can accept all of the pages contained in an imported file.

- d. Click **OK**.

The **Document Page Properties** dialog box closes.

7. From the Menu bar, select **File > Save**.

Creating a Function to Drive Automatic Page Import

Automatic page import uses a built-in function called `LiveExternalDocImport` that is available in Design Manager. You must create a custom function that uses `LiveExternalDocImport` as part of its logic.

To create a function to drive automatic page import:

1. In Design Manager, in the Library, right-click the **Functions** heading and select **New Function**.

The **New Function** dialog box opens.

2. In the **Name** box, enter a name. In the **Description** box, enter a description (optional).

3. Click **Finish**.

The function opens in the Property Panel for you to define.

4. Define the properties for the function:

- a. In the **Type area**, select **Function**.

- b. From the **Data Type** drop-down list, select **Boolean**.

- c. Define other properties as needed.

5. Specify the logic for the function.

- a. In the function **Logic** box, type value = and then click .
The **Select Built-in Function** dialog box opens.
 - b. From the **Filter** drop-down list, select the type of built-in function you want to add.
 - c. From the built-in function list, select **LiveExternalDocImport**.
 - d. Click **OK**.
The built-in function is inserted into the logic.
6. From the Menu bar, select **File > Save**.

For information about using built-in functions in Live documents, and for specific information about the arguments used in the `LiveExternalDocImport` function, see [“LiveExternalDocImport” on page 394](#).

Assigning a LiveEditor Event That Will Trigger the Import Function

In order for the new function you created to import pages automatically, you must set a LiveEditor event that will trigger the action whenever an end user performs a specified task. For example, you can specify that pages will be imported into the placeholder document when an end user submits the Live document for processing.

For information about assigning LiveEditor events, see [“Executing Functions During LiveEditor Events” on page 132](#).

As you specify arguments for the `LiveExternalDocImport` function and then assign events to trigger the function, keep in mind the following behaviors:

- If the `Append` argument is set to update the external pages, and the path specified in the `FilePath` argument does not change, then the event will trigger the function *only* the first time that it occurs. This is true even if the DOCX, DXF, PDF, or TIFF file is updated before an end user subsequently submits a DLF for processing. If a user needs to resubmit a DLF for processing, you must update the `FilePath` argument if you want an updated file to be imported into a placeholder document.
- In a similar scenario, if the `Append` argument is set to add external pages to a placeholder document, then a new set of pages will be added each time a user submits a DLF for processing. For example, if you configure the placeholder document to append a six-page PDF when a user submits a DLF for processing, then an additional set of six pages will be added to the placeholder document each time the user submits the DLF. The first submission will result in a six-page placeholder document, the second submission will result in a 12-page placeholder document, and so on.

6.3.3 Allowing End Users to Import External Pages into a Live Document

Manual document import lets end users select the external DOCX, DXF, PDF, or TIFF files that are imported into a placeholder document. Because end users will be required to select a placeholder document before they select the external files that will populate it, you should use specific, descriptive names for each placeholder document in your design. For example, a name such as "PDF Placeholder Document" is probably not specific enough for most end users. Names such as "3-Page Contract Extension" or "Ohio Policy Brochure," on the other hand, provide end users with enough information to choose the correct placeholder document to hold the pages they will import.

LiveEditor users will follow these basic steps to import external documents into a Live document:

1. From the LiveEditor file menu, users will select **Live > External pages> Place external pages**.

The **Select a placeholder document** dialog box opens.

2. From the **Select an external document** dialog box, users will select from a list of placeholder documents that will hold the imported file. This list is populated with the names of the placeholder documents that are included in the design of the DLF file but have not yet been used. Any placeholder documents whose variables do not have a value that will be populated in the engine, including empty or incorrect placeholder variables, will appear in this list.

After the user has made a selection from the list of placeholder documents, an **Open** dialog box opens.

3. From the **Open** dialog box, users will browse to the location of a DOCX, DXF, PDF, or TIFF file (as dictated by the placeholder document that they chose in step 2). After they locate the proper file and click **Open**, the placeholder document will appear in the Outline Viewer. The icons for placeholder documents are outlined in red to differentiate them from other documents in the DLF file.

If you, the designer, configured the placeholder document to accept multiple pages, each page of the external file will appear as a separate page within the placeholder document.

To set up a placeholder document that allows LiveEditor users to import external files manually, you must complete the following tasks:

1. ["Creating a New Placeholder Variable" on the next page](#)
2. ["Creating a New Placeholder Document in Design Manager" on the next page](#)

Creating a New Placeholder Variable

You must create a variable that defines the properties of the external file(s) that will be imported into the Live document. The placeholder must use the following settings:

- The **Type** must be **Placeholder**.
- The **Placeholder Type** must be one of the following options:
 - **DXF**
 - **PDF**
 - **TIFF (B&W G4 or uncomp)**
 - **TIFF (Color)**
 - **Word (*.docx)**
- If the imported file contains multiple pages, you must select the **Array** check box.

You can define the other variable properties as needed.

Creating a New Placeholder Document in Design Manager

You must create a placeholder document that will hold the content from the external files being imported. This document will use the newly created placeholder variable to define the type of external file that it can accept.

To create a new placeholder document in Design Manager:

1. Create a new document.
For information about creating documents, see *Designing Customer Communications* in the Exstream Design and Production documentation.
2. On the **Basic** tab of the document properties, from the **Document type** drop-down list, select **Placeholder (use pre-composed content)**.
3. In the **Placeholder** box, define the variable used to import the DLF.
 - a. In the **Placeholder variable** box, click  .
The **Select Variable** dialog box opens.
 - b. Browse to the placeholder variable you created for the external file.
 - c. Click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Placeholder variable** box.

4. Define the other document properties as needed.
5. Add a page to the document to make it structurally valid. The name you assign to a page will be applied to all pages imported into that document in LiveEditor.

Tip: If you want to include additional content for the customer, add those pages to the document or to a different document as needed.

For information about adding pages and defining general document properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

6. If the document will contain multiple pages, you must configure the page object to accept multiple pages.
 - a. From the Library, drag the document to the Edit Panel.
A graphical representation of the document appears.
 - b. In the right column, double-click the page name.
The **Document Page Properties** dialog box opens.
 - c. From the **Position of page in document** drop-down list, select **Filler Page (as Required)**.
This option allows the page to duplicate itself so that it can accept all of the pages contained in an imported file.
 - d. Click **OK**.
The **Document Page Properties** dialog box closes.
7. From the Menu bar, select **File > Save**.

6.3.4 Setting Up a Placeholder Document to Allow Imported DOCX or DXF Files to Flow

When a DOCX or DXF file is imported, content can overflow the end of the pages in the placeholder document if the document contains variables. By default, imported DOCX or DXF files are limited to importing content based on the original page layout and design.

Variables can affect the page layout of the imported file because when variable content is added during the engine run, the variables can sometimes add more text than can fit within the space that is provided in the original design. For example, suppose that you have a policy document

that fills three pages from top to bottom before the variables are populated. The variables that are used in the document contain updated clauses from the legal department. When these clauses are added to the document, they could add another page of text.

If the placeholder document is set up to allow the imported DOCX or DXF file to flow, the engine can add additional pages to accommodate overflow that is caused by the addition of variable content to ensure that no content is lost when the file is imported.

For more information about including variables in external files, see *Importing External Content* in the Exstream Design and Production documentation.

To set up a placeholder document to allow an imported DOCX or DXF file to flow, you must complete the following tasks:

Do this	Description
Include the ALLOW_PLACEHOLDER_CONTENT_TO_FLOW switch in the control file.	<p>When you prepare for production, you must include the ALLOW_PLACEHOLDER_CONTENT_TO_FLOW switch in the control file. The ALLOW_PLACEHOLDER_CONTENT_TO_FLOW switch allows the engine to repaginate the imported DOCX or DXF file at run time as variable content is added and as the content is placed within the pages of the placeholder document.</p> <p>To indicate that you want the engine to repaginate the imported DOCX or DXF file, you must enter YES as an argument for this switch.</p> <p>For example:</p> <pre>-ALLOW_PLACEHOLDER_CONTENT_TO_FLOW=YES</pre>
For DXF files, you must set the placeholder page that is included in the placeholder document to flow to a specific flow page.	<p>In LiveEditor, the flow of imported documents are controlled by the setup of the placeholder document. To allow the content of the imported DXF file to flow, you must set the placeholder page in the placeholder document to flow to a specific flow page. Any flow settings that were previously applied to the DXF file are ignored.</p> <p>To set a placeholder page to flow to a specific flow page:</p> <ol style="list-style-type: none"> 1. In Design Manager, open the placeholder page in the Property Panel. 2. Click the Flow tab. 3. From the Destination of overflow from this page drop-down list, select Flow to specified page. 4. From the Page drop-down list, select the flow page that you want to use to accommodate overflow from the imported DXF file. <p>For more information about setting up flow pages to accommodate flowing content, see <i>Designing Customer Communications</i> in the Exstream Design and Production documentation.</p>
For DXF files, make sure that the files that end users can import do not contain unique flow pages or complex flow settings.	<p>The flow of DXF files that are imported into LiveEditor are controlled by the placeholder document (not the flow settings originally included in the DXF file). Since the placeholder document can only be set to flow to a single flow page, you must make sure that the files that end users can import do not contain unique flow pages or complex flow settings. Allowing end users to import files with complex flow settings or unique flow pages can produce unexpected results.</p>

Do this	Description
For DXF files, validate the flow and relativity settings in the original file.	<p>As the engine paginates the imported content at run time, grow and relativity settings that were applied to the original DXF file content can affect the appearance of the DXF file in the final customer output. Objects within a DXF file behave as they were originally designed, even as content is repaginated to accommodate variable text. For example, suppose that a DXF file contains a text box, and beneath the text box is a table. If the Move relative to the object option for the table is set to Above, the table will be pushed down the page as the text box grows to accommodate the added variable text. If the Move relative to the object option for the table is set to None (does not move), the text box can grow and overlap the table in the design. To ensure that the DXF file flows as expected after the engine adds variable content to the final customer output, validate the flow and relativity settings in the original DXF file.</p> <p>For more information about flow and relativity in DXF file designs, see <i>Designing Customer Communications</i> in the Exstream Design and Production documentation.</p>

Note: When you import files at run time, Exstream can repaginate only DOCX or DXF files to prevent overflow that is caused by variable content or different page sizes. If you want to allow documents in other formats (such as PDF, InDesign, or Quark) to flow when they are imported at run time, you can use the Exstream conversion tools to convert external document formats to the DXF format before importing the document at run time.

For a full list of formats that can be converted to the DXF format and for more information about using the Exstream conversion tools, see *Importing Designs* in the Exstream Design and Production documentation.

Chapter 7: Allowing End Users to Add and Modify Copies of Documents

If you are using Exstream Live to create correspondence or other documents that must be completed by end users and sent to recipients or destinations in addition to the customer, you can allow end users to add copies (known as "recipient copies"). For example, suppose you use LiveEditor to let end users complete insurance claim letters. The claimant (the customer) will receive the letter complete with terms and conditions, and you might allow end users to send the following additional copies of the letter:

- For an attorney, a copy with the same documents as the claimant
- For the claimant's agent, a copy without terms and conditions
- For long-term retention in an archive, a copy with only the legally required information

You can design the Live document so that end users can edit the original claim letter for the claimant and then add the appropriate copies that automatically include the necessary documents for the additional recipients.

When working with recipient copies, keep in mind that the term "original document set" refers to the set of documents in the application that is normally sent to the customer. If it is appropriate for your application, you might instead send a recipient copy to the customer, but this is an uncommon arrangement. In the above example, the original document set is sent to the claimant.

Just as in traditional Exstream Design and Production applications, you use recipient profiles to configure different types of recipients. For Live documents, you can also create recipient data pages that allow end users to add or update information about recipients, such as names and mailing information. When adding a recipient copy, the end user selects from the recipient profiles you set up, and then copies of the documents indicated by the recipient profile are added to the Live document, as well as cover pages and recipient data pages if they are included in the recipient profile. If you include a recipient data page, the end user then enters the recipient's mailing or other information for each new recipient copy.

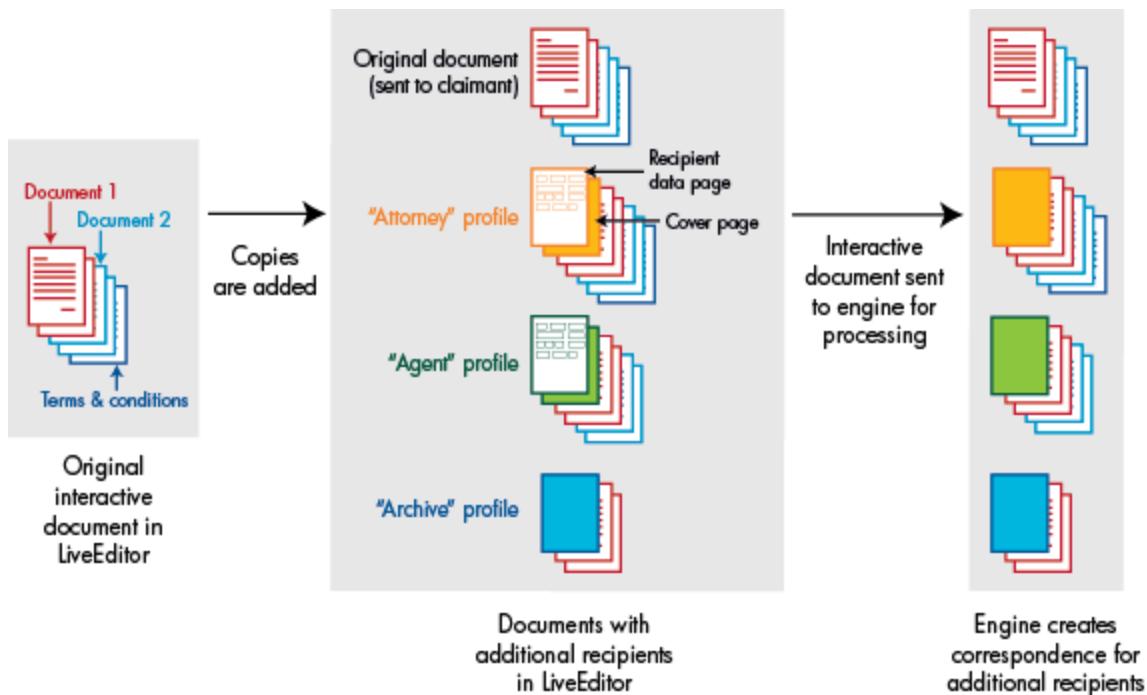
Alternatively, you can add recipient copies automatically by storing the information of the additional recipients for each customer in recipient data records in a data file, as you would in traditional Exstream Design and Production applications. Continuing the above example, suppose the information for each claimant's agent is managed by an external system. You can store agent names and mailing or other information in a data file, and appropriate recipient copies are added from the recipient records automatically when the end user opens the Live document in LiveEditor.

For more information about adding recipient copies automatically, configuring recipient profiles, and mapping recipient data records in data files, see *Designing Customer Communications* in the Exstream Design and Production documentation.

The following graphic illustrates how an end user can add recipient copies using the recipient profiles from the above example. Each recipient profile specifies a cover page tailored to the type of recipient, and the recipient profiles for recipient copies that will need recipient information added or edited by the user include recipient data pages. The following document sets are shown:

- The original documents ("Document 1," "Document 2," and "terms and conditions") are sent to the claimant.
- The "attorney" profile specifies the same document set as the original.
- The "agent" profile removes the "terms and conditions" document.
- The "archive" profile removes "Document 2" and the "terms and conditions" document so that only "Document 1" is included for archiving. The "archive" profile does not include a recipient data page because it uses the existing customer information.

Illustration of adding recipient copies to a document



Note that a recipient profile can only remove documents from an application; it cannot add a document from outside the application, or reorder the documents in the application. In other words, documents included in a recipient profile appear in recipient copies only if they are included in the application that is using the recipient profile, and they appear in the same order as they appear in the application.

In LiveEditor, an end user can add recipient copies by using the **Live** menu. You can also design a Live document to add recipient copies automatically based on an action using the **AddRecipient** function, or using recipient data records in a data file. Regardless of the method by which recipient copies are added, when a copy of an original document or documents is

added to the Live document, the document copy is added after the last page of the original documents for each customer, and is preceded by a recipient data page and a cover page, if included. The recipient data page is displayed immediately if it is included so that end users can provide the recipient's information.

To allow end users to add and modify recipient copies, complete the following tasks as needed:

1. If you want to allow end users to modify recipient information, create a recipient data page.
2. If you want to use cover pages for recipient copies, create a cover page for each recipient type.

For information about creating cover pages, see *Designing Customer Communications* in the Exstream Design and Production documentation.

3. Create at least one recipient profile. If you want to manage multiple recipient types, create the appropriate additional recipient profiles.

For information about creating recipient profiles, see *Designing Customer Communications* in the Exstream Design and Production documentation.

4. Configure recipient profiles for your Live document.
5. Add the appropriate recipient profiles to your application.

For information about adding recipient profiles to your application, see *Designing Customer Communications* in the Exstream Design and Production documentation.

6. If you want to include the capability to add recipient copies based on an action, set up the relevant action.
7. If you want to populate recipient data from outside LiveEditor, map recipient data records in a reference file.

If you do not want to allow end users to add recipient copies at all in a particular Live document, do not add any recipient profiles to the application. If you want to hide the **Recipients** menu in LiveEditor, use the view object properties to disable the recipient options on the **Live** menu.

For information about setting up a view object, see “[Using Views to Customize the End-User Interface](#)” on page 226.

7.1 Creating Recipient Data Pages to Allow End Users to Modify Recipient Information

A recipient data page is a basic page object that allows end users to edit recipient information using fields or other objects tied to variables. A recipient data page is often set up similarly to an interview page. For example, you might set up fields in which end users can enter a recipient's name, email address, and mailing address. You then add the recipient data page to a recipient profile. When the end user adds a recipient copy using that recipient profile, or when a copy is

added automatically using that recipient profile, the recipient data page precedes the recipient copy in the Live document. Because recipient data pages are used only to collect data, they do not appear as part of the recipient copy when the document is saved as a PDF file, printed, or processed by the engine.

Example of a recipient data page

Enter recipient information here:

First name

Last name

Mailing address line 1

Mailing address line 2

City

State

ZIP code

Email address

To create a recipient data page:

1. Create a new page as you normally would in Designer.
2. Use the design tools to set up the page as needed. You can use the form controls and selection tools to set up the page so that end users can easily enter the recipient information.
3. Add the recipient data page to a recipient profile to use it in the recipient copy process.

For general information about creating pages or using the design tools in Designer, see *Designing Customer Communications* in the Exstream Design and Production documentation.

For information about using selection controls to design a Live document, see “[Controlling End User Editing Using Selection Controls](#)” on page 76.

For information about adding a recipient data page to a recipient profile, see “[Configuring Recipient Profiles for Live Documents](#)” on the next page.

Variables used on a recipient data page affect only that recipient's copy of the documents. When you are using recipient data records in a reference file or customer driver file, changes made by end users on a recipient data page are also made in the recipient data in the file. For example, suppose the 'Customer_First_Name' variable is used in the original document. After a recipient copy is added to the Live document, an end user can interact with a field on the

recipient data page to update the 'Customer_First_Name' variable value for that recipient's copy of the document. This updates any occurrence of the 'Customer_First_Name' variable within the recipient copy to reflect the recipient's unique information, as well as the value of the variable in the recipient's record in the data file. However, other occurrences of the 'Customer_First_Name' variable, either in the original document or other recipient copies, are not updated.

For information about using recipient data records in data files with Live documents, see ["Using Recipient Data Records to Populate Recipient Data From Outside of LiveEditor" on page 184](#).

For general information about mapping recipient data records in data files, see *Designing Customer Communications* in the Exstream Design and Production documentation.

A recipient data page is not required. When you use a data file with recipient data records without a recipient data page, recipient copies use the external recipient data and cannot be changed by end users. If you allow end users to add recipient copies but you don't include a recipient data page or a data file with recipient data records, data in recipient copies cannot be modified, and documents in recipient copies are exact copies of the original.

For general information about managing recipient copies, see *Designing Customer Communications* in the Exstream Design and Production documentation.

The properties you set for the recipient data page theme affect how the recipient data page appears in LiveEditor. Consider making the theme for recipient data pages unique so that end users can easily see which pages will not appear in the final output.

For information about setting up themes, see ["Using Themes to Visually Guide End Users" on page 218](#).

7.2 Configuring Recipient Profiles for Live Documents

Just as you create recipient profiles to manage different types of recipients in a traditional Exstream Design and Production application, you can use recipient profiles in a Live document to allow end users to add copies for different types of recipients or destinations. When adding a recipient copy, an end user selects from the recipient profiles you include in the Live document to determine the type of recipient or destination to which the copy should be sent.

If you set up the Live document to automatically add recipient copies with recipient data records in a data file, the recipient profiles used can be determined by the recipient data records in a data file, just as in a traditional Exstream Design and Production application. The recipient profile object properties for Live documents that are listed in the table below still apply to recipient copies added automatically in LiveEditor.

For information about creating recipient profiles, see *Designing Customer Communications* in the Exstream Design and Production documentation.

A recipient profile contains settings that specifically apply to Live documents, such as the recipient data page used for each copy and whether end users can print each recipient copy.

For example, suppose you want to send a claim letter to a claimant and send a copy to the claimant's agent. You might create an "agent" recipient profile and include a recipient data page to have the end user enter the name and mailing information of the claimant's agent.

To enable recipient copies, you must create at least one recipient profile.

Use the following options on the recipient profile object properties to configure the profile for Live documents:

To	Do this
Include a recipient data page to allow the end user to edit recipient data for each copy that uses this recipient profile	<ol style="list-style-type: none"> Click the Recipient data page box. The Select Page dialog box opens. Select the page you want to use and click OK. The Select Page dialog box closes and the page you selected appears in the Recipient data page box. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: If you are not using pre-populated recipient data from a data file and you do not add a recipient data page, then when an end users adds a recipient copy, the documents in the copy are exact replicas of those in the original document set; that is, no data in the copy, such as mailing information, will be customized in the copy. </div>
Allow end users to print copies that use this recipient profile from LiveEditor	Select the Recipient page copies can be printed check box. If you clear this check box, end users receive a message if they try to print a recipient copy that uses this profile.
Allow end users to save copies that use this recipient profile as PDFs from LiveEditor	Select the Recipient page copies can be saved as PDF check box. If you clear this check box, then when end users save the Live document as a PDF file, any recipient copies that have been added to the Live document using this profile are not included in the PDF.
Prevent variable values from being changed by recipient data in a data file or by end users when a copy is added using this recipient profile in LiveEditor	Select the Variable values are preserved from original document in LiveEditor check box. When this option is selected, the Recipient data page box is unavailable, and no recipient data page is shown with the recipient copy in LiveEditor. Variables values are preserved from the original document; that is, documents in the copy are exact replicas of those in the original document set even if you are using a data file with recipient data records.

7.3 Including the Capability to Add Recipient Copies Based on an Action

If you want to automatically add recipient copies based on an event, use the `AddRecipient` built-in function in your design. For example, you might design a button labeled "Create an archival copy" and associate the `AddRecipient` function with it. When end users click the button, a recipient copy is added using the recipient profile specified with the function. If the

recipient profile used includes a recipient data page, end users can then provide the recipient information as needed.

For more information about setting up the AddRecipient function, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

7.4 Including Recipient Copies in Multiple Languages

You can use the 'SYS_LanguageCustomer' variable to specify the language used for a recipient copy. For example, you can include the value of the variable as part of the recipient data page, or you can map the data file containing the recipient data records to the value of the variable. In either case, the language used in the recipient copy will be independent of the language used in the parent customer document. If you do not associate the 'SYS_LanguageCustomer' variable with the recipient data, then the language of the recipient copy will be the same as that of the parent customer document.

For more information about using multiple languages in a Live document, see [“Designing Live Documents for Multiple Languages” on page 68](#).

7.5 Using Recipient Data Records to Populate Recipient Data From Outside of LiveEditor

If you want to populate recipient data from outside of LiveEditor, you can map recipient data records in a data file just as you would in traditional Exstream Design and Production applications. This method allows data for recipient copies to be managed by an external system and optionally predefined for an end user. For example, you might store the name and mailing information for customers' agents and case managers in an external system. Using an external process, you might populate recipient records in a data file with this information. When an end user opens the Live document in LiveEditor, recipient copies are automatically added using the recipient data records from the data file.

For more information about mapping recipient data records, see *Designing Customer Communications* in the Exstream Design and Production documentation.

If no data file is used, end users can still add recipient copies and, if you provide a recipient data page, edit recipient data, but recipient data cannot be changed outside LiveEditor.

A data file used for recipient data in a Live document must be an XML formatted reference file, which requires you to license the XML/JSON Input module. To use the recipient data in a Live document, associate the reference file with the setting object you will use to create the Live

document. Since you are using a reference file, keep in mind that on the **Basic** tab of the data file properties, you must select **Driver-ordered, required** or **Driver-ordered, optional** from the **Access** drop-down list.

For more information about XML formatted reference files or the XML/JSON Input module, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To associate the a reference file containing recipient data records with a Live document:

1. In Design Manager, from the Library, drag the setting object you will use to create the Live document to the Property Panel.
2. Click the **Data** tab.
3. Click the **Recipient data reference XML file** box.

The **Select Reference File** dialog box opens.

4. Select the data file to use for recipient data and click **OK**.

The **Select Reference File** dialog box closes and the data file you selected appears in the **Recipient data reference XML file** box.

5. From the Menu bar, select **File > Save**.

Alternatively, you can include recipient data within the customer driver file as section data. However, recipient data stored in the customer driver file can be edited only within LiveEditor after the Live document is created; changes directly within the customer driver file are not recognized.

For more information about customer driver files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

7.6 Processing Considerations for Live Documents with Recipient Copies

If you are using recipient data records in a data file, you must include data in the DLF output. On the **Basic** tab of the DLF output properties, select the **Include data** check box.

For more information about setting up a DLF output object, see “[Setting Up the Objects Necessary for Live Output](#)” on page 354.

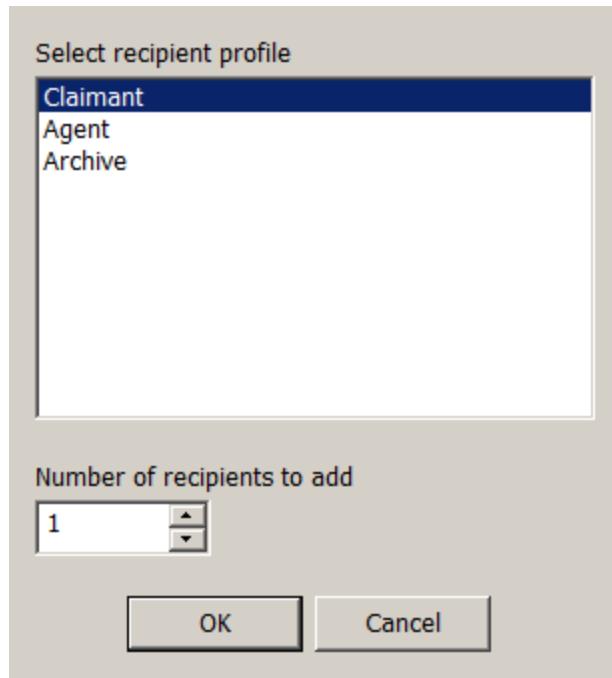
7.7 Adding Recipient Copies: The End User Experience

This section describes how end users will add recipient copies in LiveEditor.

1. An end user selects a document from the Outline Viewer and then selects **Live > Recipients > Add Recipient** from the Menu bar.

The **Add Recipient** dialog box opens.

Add Recipient dialog box



2. The end user selects the number of copies to add and, if you included multiple recipient profiles in the application, selects a recipient profile to use for the copies.

A recipient copy is added using the selected profile.

3. If you included a recipient data page for the selected profile, the recipient data page appears in the design window, and the end user completes the recipient data page.

Example of a recipient data page

Enter recipient information here:

First name	Last name	
<input type="text"/>	<input type="text"/>	
Mailing address line 1	Mailing address line 2	
<input type="text"/>	<input type="text"/>	
City	State	ZIP code
<input type="text"/>	<input type="text"/>	<input type="text"/>
Email address		
<input type="text"/>		

4. The end user can add copies using another recipient profile by repeating step 2 through step 3.

After adding recipient copies, end users can use the options on the **Live > Recipients** menu to duplicate an individual copy, select a different recipient profile to use for an individual copy, or remove an individual copy.

Documentation for adding recipient copies is included in the integrated help feature of LiveEditor. However, to help end users understand the specific process for your design, you might want to provide end users with your own documentation.

Chapter 8: Using a Web Service to Exchange Data Between Live Documents and Other Enterprise Systems

One way to leverage the content that is stored, created, and edited in Live documents is to collect the content and use it in other enterprise applications (separately from the Live document). For example, suppose you use a Live document as an account enrollment form. After the business user fills out the document with the new enrollee's information, you can collect the data and update your customer database with the new information. This process is carried out by end users submitting the content by way of a web service. Live document content that is transmitted by a web service can either be sent inline or as an attachment, giving you the flexibility to design the application in the way that best integrates with your existing systems. You can also use a web service in conjunction with formatted text variables to collect, store, and leverage formatted text.

Note: Double-byte content submitted by way of a web service is sent in UTF-16 little-endian format.

To use a web service to gather Live document content, you must have licensed the Web Services Interface module. When you license the LiveOutput module, you automatically have access to the functionality included with the Web Services Interface module.

Keep in mind that if you are using data files to communicate with a web service in a Windows environment and you do not plan to install the design environment, you must install the Microsoft Visual C++ 2013 Redistributable Package.

For more information about installing the production environment on Windows, see *Installation and Upgrade Information* in the Exstream Design and Production documentation.

To use a web service to leverage Live document content, complete the following tasks as needed:

- “[Designing the Live Document to Support Data Submission](#)” on the next page
- “[Mapping the Request Data](#)” on the next page
- “[Mapping the Response Data](#)” on page 191
- “[Setting Up Success and Error Messages to Appear in LiveEditor After Web Service Calls](#)” on page 191
- “[Using Single Sign-On Authentication for Live Documents Running on a Web Service](#)” on page 193

8.1 Designing the Live Document to Support Data Submission

As with any Live document, you should use the tools and features available in Exstream Design and Production to design the Live document so it is easy and intuitive for end users to use. For example, you might consider creating a custom button that is placed on a design page so end users can click it to submit the Live document's data to the Web service.

For more information about buttons and other features you can use in Live document to support the editing experience, see the [“Using Live Tools to Create a More Intuitive and Automated Editing Experience” on page 123](#).

8.2 Mapping the Request Data

You set up a data file that is mapped to the data that will be submitted to the web service. This data is called the request data. As you set up the data file for the request, keep in mind the following requirements:

- If you are requesting data and you want to receive data from the web service response, you must map the request data in an auxiliary layout file using the web service (SOAP) format or the web service (RESTful) format, depending on whether you are using a SOAP or RESTful web service.
- If you are submitting data and you do not expect data from the web service response, you must map the request data in a report file or a post-sort report file using the web service (SOAP) format or the web service (RESTful) format, depending on whether you are using a SOAP or RESTful web service.
- You must map the entire request message (the body and header, if used).
- Depending on the content from the Live document that you want to send to the web service, you must use one of the following system variables in the mapping:
 - **'SYSLD_Content'**—Use this system variable if you want to send the full contents of the DLF to the web service.
 - **'SYSLD_ContentXML'**—Use this system variable if you want to send only the content XML from the Live document to the web service.

Note: You cannot use both 'SYSLD_Content' and 'SYSLD_ContentXML' in the same data file.

Simple example of mapped request

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http:
<soap:Body>
  <Payment>True</Payment>
  <DLF>cid:8213836380435</DLF>
</soap:Body>
</soap:Envelope>
```

In this example, cid:8213836380435 is mapped to 'SYSLD_Content'.

For more information about creating and defining data files for web services, see ["Using a Web Service to Exchange Data Between Live Documents and Other Enterprise Systems" on page 188](#).

For more information about mapping data files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

For more information about the 'SYSLD_Content' and 'SYSLD_ContentXML' variables, see ["Live System Variables" on page 426](#).

8.2.1 Submitting Content Using Attachments

If you want Exstream to submit Live document content or content XML to a web service as an attachment instead of inline, after mapping the request data, you must complete the following additional steps:

1. With the auxiliary layout file in the Edit Panel, right-click the mapped data area and select **Data Area > Data Area Properties**.

The **Data Area Properties** dialog box opens.

2. Click the **Attachment** check box.
3. Click **OK**.
4. Save the auxiliary layout file.

5. From the Edit Panel, drag the auxiliary layout file to the Property Panel.

The **Attachment handling** area becomes active.

6. From the **Attachment method** drop-down list, select one of the following options to specify the method Design Manager uses to attach the files sent to the server:
 - **MTOM/XOP**—Attaches the DLF or customer XML using the Message Transmission Optimization Mechanism method. This method requires key words to identify internal attachments.
 - **SWA cid scheme**—Attaches the DLF or customer XML using the SOAP With Attachments Content ID (SWA CID) scheme. This method uses key words to identify internal attachments. Attachments without CIDs are assumed to be external.

- **SWA**—Attaches the DLF or customer XML using the SOAP with Attachments (SWA) method.
7. From the **Attachment encoding method** drop-down list, select one of the following options to specify how Design Manager encodes attachments:
- **Binary**— Design Manager includes attachments in their binary format. The attachments are not encoded.
 - **Base-64**— Design Manager encodes attachments using Base-64 encryption.
8. From the Menu bar, select **File > Save**.

8.3 Mapping the Response Data

The response data is the data that is received from the web service. Just as with the request data, you must map a data file to set up the response data. As you set up the data file for the response, keep in mind the following requirements:

- You must map the response data in a reference file using the web service (SOAP) format or the web service (RESTful) format, depending on whether you are using a SOAP or RESTful web service.
- You must map the entire response (the body and header, if used).
- If you do not expect response data from the request, you do not need to map response data.

Simple example of mapped response

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<update>document updated</update>
</soap:Body>
</soap:Envelope>
```

8.4 Setting Up Success and Error Messages to Appear in LiveEditor After Web Service Calls

You can set up custom success and error messages that appear in LiveEditor after Live document data has been submitted or failed to be submitted to a web service during a call. If you

do not set up a custom error message, LiveEditor does not issue an error message if there is an error. Error messages you set up can contain one or more of the following components:

- **Error component**—A general description of the error
- **Custom component**—Custom information about the error, such as a string variable that provides a link or email address to contact for error messages
- **Detail component**—Specific information about the error

Tip: If you include system information in the detail component, it might contain information that is generally not user-friendly, but is helpful for troubleshooting. If you provide contact information within the custom component, consider providing the system-generated information in the detail component so end users can provide it to the point of contact.

To set up success and error messages that appear in LiveEditor after web service calls:

1. Create and map a web Service (SOAP) or web Service (RESTful) data file.
For more information about creating and defining data files for web services, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.
2. From the Library, drag the data file to the Property Panel.
3. Click the **Interactive** tab.
4. To specify the message issued for a successful web service call, select one of the following options from the **LiveEditor Web Service Success Message** drop-down list:

To	Do this
Issue no message	Select None .
Issue a message with the information specified by a user-defined variable	Select Variable , and from the box below the drop-down list, select a string variable.

5. To specify the type of information you want to appear in the error component of an error message, select one of the following options from the **Error component** drop-down list:

To	Do this
Issue no information in the component	Select None .
Issue system-generated information	Select System .
Issue information specified by a user-defined variable	Select Variable , and from the box below the drop-down list, select a string variable.

6. To specify the type of information you want to appear in the custom component of an error message, select one of the following options from the **Custom component** drop-down list:

To	Do this
Issue no information in the component	Select None .
Issue information specified by a user-defined variable	Select Variable , and from the box below the drop-down list, select a string variable.

7. To specify the type of information you want to appear in the detail component of an error message, select one of the following options from the **Detail component** drop-down list:

To	Do this
Issue no information in the component	Select None .
Issue system-generated information	Select System .
Issue information specified by a user-defined variable	Select Variable , and from the box below the drop-down list, select a string variable.

8. From the Menu bar, select **File > Save**.

8.5 Using Single Sign-On Authentication for Live Documents Running on a Web Service

If you are designing a Live document that will be hosted on a web service, you can adjust the authorization settings on the Live setting object so that end users will not have to log in twice to access the Live document—first to the web service and then to LiveEditor. The authorization settings allow LiveEditor to communicate with the web service to retrieve a simple XML file that LiveEditor uses to authenticate end users according to design groups that you set up in Design Manager.

For example, if an end user logs in to a web service and then opens a Live document being hosted there, LiveEditor will then make a call to the URL specified on the Live setting object, retrieve the XML file that details the design groups that can access the Live document, and then grant editing permission to each end user according to the design groups to which he or she belongs. If the web services call fails during single sign-on authentication, then the server will return standard HTTP status codes to describe the error.

Keep in mind that single sign-on authentication is less about security than ease of use—both for you and for end users. One major benefit of using this method is that it lets you employ single sign-on authentication without having to deploy custom authentication DLL files. If you were to use custom authentication DLL files with an installed base of several thousand end users, you would not only have to deploy the DLL files to each end user's computer, but you would also have to update the DLLs as necessary, making sure that everyone is using the latest version. In contrast, you can set up a simple XML file and adjust a few settings so that you do not have to

manage custom authentication DLL files, and end users do not have to enter multiple user names and passwords to access a Live document.

To set up single sign-on authentication for Live documents running on a web service, you must complete the following tasks:

1. [“Setting Up an XML File to Identify Which Groups Can Access the Live Document” below](#)
2. [“Adjusting the Authorization Settings on the Live Setting Object” on the next page](#)

8.5.1 Setting Up an XML File to Identify Which Groups Can Access the Live Document

To set up single sign-on authentication for Live documents, you must first create an XML file that specifies which design groups will be able to access your Live document. This file must be no larger than 128KB.

Tip: When specifying which design groups will be able to access your Live document, keep in mind that the names in your XML file must match the names of stand-alone design groups and/or the names of design groups in the **Groups included in group** field in the design group properties in Design Manager.

For more information about setting up design users and design groups, see *System Administration* in the Exstream Design and Production documentation.

The following example shows how your XML should be structured:

```
<?xml version="1.0"?>
<live:Groups xmlns:live="http://www.hp.com/exstream/live"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.hp.com/exstream/live.xsd">
  <NetworkGroup name="Authors"/>
  <NetworkGroup name="Editors"/>
  <NetworkGroup name="Admins"/>
</live:Groups>
```

Your XML file is validated against the following schema, which is embedded in the `LiveEditor_xx-xx.dll` file:

```
<xsd:schema
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:live="http://www.hp.com/exstream/live"
    targetNamespace="http://www.hp.com/exstream/live">
    <xsd:annotation>
        <xsd:documentation xml:lang="en">
            Copyright (c) 2016 Open Text. All rights reserved.
        </xsd:documentation>
    </xsd:annotation>
    <xsd:element name="Groups" type="live:GroupsType">
        <xsd:unique name="NetworkGroupNames">
            <xsd:selector xpath="NetworkGroup"/>
            <xsd:field xpath="@name"/>
        </xsd:unique>
    </xsd:element>
    <xsd:complexType name="GroupsType">
        <xsd:sequence>
            <xsd:element name="NetworkGroup" type="live:NetworkGroupType"
                minOccurs="0"
                maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="NetworkGroupType">
        <xsd:attributeGroup ref="live:NetworkGroupTypeAttributes"/>
    </xsd:complexType>
    <xsd:attributeGroup name="NetworkGroupTypeAttributes">
        <xsd:attribute name="name" type="xsd:string" use="required"/>
    </xsd:attributeGroup>
    </xsd:complexType>
</xsd:schema>
```

8.5.2 Adjusting the Authorization Settings on the Live Setting Object

After you have created an XML file that specifies which design groups will be able to access your Live document, you must specify the appropriate authorization settings on the Live setting object associated with your Live document.

To adjust the authorization settings on the Live setting object:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Authorization** tab.
3. From the **User authentication method** drop-down list, select **ACL**.

The **Group access** field and the **External URL authentication** section become available.

4. In the **Group access** field, specify the design groups that will be able to access your Live document.

For more information about specifying which design groups can access a Live document, see [“Controlling Access to Live Documents and Areas within Live Documents” on page 286](#).

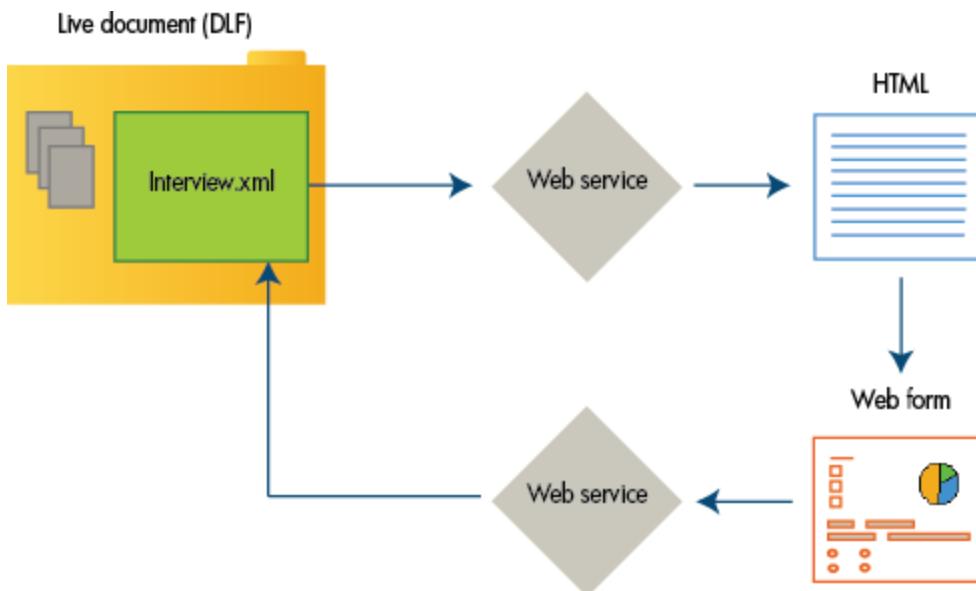
5. In the **External URL authentication** section, in the **Authentication URL** field, enter the URL for the XML file that specifies which design groups will be able to access the Live document.
6. From the Menu bar, select **File > Save**.

Chapter 9: Collecting Data For a Live Document Using Web Forms

Sometimes document life cycles require you to gather information from external sources, such as users who work outside your organization's firewall. To help meet this requirement, Exstream Live enables you to leverage its data collection features (interview pages) and integrate them into your organization's web application using a web service you set up. After data has been collected through the web form, it can then be incorporated back into the Live document and leveraged throughout the rest of the Live document workflow (for example, viewed and manipulated in LiveEditor, stored in data repositories, and published as a final deliverable through the Exstream Design and Production engine).

When you set up a Live document so that it can be presented as a web form, information about the objects on the interview page is produced in XML format and stored within the Live document when the Live document is generated. You can then set up a web service call that accesses that XML file (`Interview.xml`) so that it can be transformed into a different format, such as HTML. The fields and objects included on the interview page are presented on the webpage and end users can interact with them. After an end user has finished interacting with the web form, the captured data is returned through a web service to the Live document. Any variables affected by the collected data are updated in the Live document, just as if the data had been collected in LiveEditor.

Illustration of document life cycle using web forms



To set up a Live document so it can be used to collect data using a web form, you must create an interview page to collect data.

This chapter discusses the following topics:

- “[Creating an Interview Page to Collect Data](#)” below
- “[Best Practices for Designing a Document to Collect Data Using Web Forms](#)” on the next page

9.1 Creating an Interview Page to Collect Data

The process and features you use to set up an interview page that will be presented as a web form are the same as the process and features used to create traditional interview pages. For example, you can include objects such as drop-down lists and radio buttons to collect data from the end user. When the interview document is rendered as HTML, those objects will appear in the web form and can be interacted with just as they can be in LiveEditor. Other Live features, such as themes, will also appear in the web form.

For more information about interview page objects and the properties that are supported for use in web forms, see the `Content.XSD` schema in the Exstream installation directory.

The only difference in creating an interview page that will be used to collect data by way of a web form is that you must designate the interview page as one that will be used that way.

To create an interview page to collect data:

1. In Design Manager, from the Library, drag the page that you will set up as the interview page to the Property Panel.
2. Click the **Basic** tab.
3. If the page is not yet designated as an interview page, select the **Live interview page** check box.
4. Select the **Use interview page as Web form** check box.
5. From the Menu bar, select **File > Save**.

When the application is packaged to DLF output, the objects on the page will be rendered in the `Interview.xml` file within the DLF.

9.2 Best Practices for Designing a Document to Collect Data Using Web Forms

Although most traditional design principles apply when you create an interview page that will be used to collect data through the web, there are a few best practices and design considerations you should review before designing your application:

- Editable areas on the interview page must be associated with a content variable that is mapped in the `Customer.xml` file in order for the content to be editable in the Web form.
- If you use the show/hide feature with content selection objects, only content that is set to be initially visible in LiveEditor will be included in the `Interview.xml` file.
- Text that is set up as a hyperlink is not retained as a hyperlink in the `Interview.xml` file.
- For more information about interview page objects and the properties that are supported for use in web forms, see the `Content.XSD` schema in the Exstream installation directory.
- Consider the intended use of the Live document and its interview page as you plan your design, and adjust your design accordingly. For example, if the interview page will be used in both LiveEditor and as a web form, you might choose to create two versions of the interview page: one optimized for viewing in LiveEditor and one optimized for conversion to a web format. The decision to create two versions of the interview page will often be driven by the types of objects you want to include on the page and their level of support in the XML version of the interview page. If you do choose to create two interview pages, consider the following design recommendations for an interview page intended for use on the web:
 - Add metadata to objects during the design to enable additional functionality when the XML is converted.
 - If you are designing a page with complex layout, such as many objects that must be aligned, consider using tables and embedding objects within cells to help retain the appearance you want.

Chapter 10: Using Section Data with Live Documents

Section data is a data layout that uses groups of records related to one particular type of information. It is a powerful feature you can use to design section objects, documents, and tables a single time and then reuse them as required based on data. For example, suppose you are setting up monthly statements for a mobile phone company in a traditional Exstream Design and Production application. The data for an account that uses a family plan may contain groups of records for each device on the account. By using section data, you can design a single table—complete with headers, footers, and call details—that repeats as necessary on the statement for each phone.

The basic principles of using section data with Exstream Live are the same as the principles for using section data in traditional Exstream applications. However, there are some differences, both in the way section data is processed and in the types of features that are supported with section data. In addition, you must consider how section data affects the way end users' changes appear and behave.

This chapter discusses the following topics:

- “[An Introduction to Using Section Data in a Live Application](#)” below
- “[How Section Data Affects Editing](#)” on the next page
- “[Live Feature Support with Section Data](#)” on page 203

For general information about section data principles, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

10.1 An Introduction to Using Section Data in a Live Application

In traditional Exstream applications, section data-driven objects are created when the engine processes the data sections and generates objects—such as tables or documents—as needed. In Exstream Live, however, the section data is included as part of the DLF file in the underlying XML. During editing, the section data file is used as a basis for driving new objects, and changes made during editing are stored in the underlying XML.

In a Live application, section data-driven objects are controlled in one of two ways:

- **Functions**—Built-in functions such as AddDataSection and DeleteDataSection allow end users to add or remove objects driven by underlying data sections. New objects are based on the data section in which the function was fired.

For more information about the built-in functions you can use to manipulate section data, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

- **XML addition**—Objects driven by section data can be added when XML containing customer information in section data format is added to the Live document (by way of a built-in function or other method).

When end users interact with objects based on section data, they can see immediate changes to objects they manipulate. For example, if end users execute a function to create a new document, the data section is updated appropriately and the new document appears in the design. Because both the visible changes to the Live document and the changes to the underlying section data are immediate, end users can print the Live document or save it as a PDF, and it will reflect the latest changes to the data.

You can also use section data to drive the content of placeholder documents. For example, suppose you work for a financial services company and are designing a Live document that will serve as a statement for each of your company's customers. Each statement must include a prospectus for new mutual funds that customers have purchased since receiving their previous statement. In that case, the content of the placeholder document could be tied to a variable that is driven by repeating data sections in the customer driver file. Section data allows the value of this placeholder variable to be unique for each mutual fund, so that the variable can drive placement of a different prospectus each time a data section is encountered.

For information about including placeholder documents in a Live document, see [“Allowing End Users to Import/Export Components of a Live Document” on page 156](#).

When you use section data in a Live application, remember that LiveEditor stores all of the section data in memory. Storing the data in memory allows LiveEditor to access all the data section data throughout the editing session; however, it may affect performance adversely when editing large documents.

10.2 How Section Data Affects Editing

The most common way to add section data-driven objects in a Live document is to use a function tied to a button or other Live object. Since this method is significantly different from traditional section data implementation, keep the following principles in mind when you set up the section data and the application:

- If an end user executes a function to add a section data-driven object, the function is run in the context of the current section data. In other words, a function fired in a data section affects only other objects driven by that data section. For example, if a function is run to update text inside a data section-driven document, only text in documents driven by the same data section is affected.
- Rules are run in the context of the current section data. Specifically, rules use the variable values of the current data section.

- If an end user deletes an object driven by section data, the underlying data is not lost. However, the object will no longer appear in LiveEditor.
- The default number of elements in an array variable is zero. Therefore, a growing array variable will be empty until elements are assigned to it. If the array variable remains empty, it will not contain any editable content, and it will not drive automated content, such as table rows or paragraphs.

Tip: You can use the function that adds the data section to provide the value of the array's first element.

- For paragraph objects, if end users make edits to an interactive area, those edits are also applied to any other instance of the same paragraph object (regardless of whether that paragraph object is included in a section objects that is driven by section data), because the paragraph object is a single design object that is used in multiple locations.

If you want to allow end users to make changes to a paragraph that are unique to other instances of the same paragraph, you must include the paragraph object using a data section driven section object. In the data, you would map a variable in the same data section that drives the inclusion of the section object. In the paragraph object, you would include a reference to that variable. End users could then edit the variable and their changes would only affect that single instance of the paragraph object in the data section driven section object.

- If a variable is included in an object driven by section data but is not included in the section data, and an end user changes the value of the variable, those values are changed throughout the customer (not just in the object driven by the data section). On the other hand, if a variable is in a data section, changes to that variable affect only other objects driven by that data section and not every instance of that object throughout the customer.
- If end users make a change to objects driven by customer-level data, only the current customer is affected. For example, if a data section is added to drive a section, then only the section in the current customer is affected by that change.
- Because the underlying section data determines the order in which section-driven documents appear in a Live document, section-driven documents cannot be moved using the Outline Viewer panel in LiveEditor.

For information about allowing end users to reorder documents in LiveEditor, see [“Allowing End Users to Reorder Documents Using the Outline Viewer” on page 58](#).

- Placeholder documents that contain section data behave differently from standard placeholder documents in the following ways:
 - Normally, placeholder documents whose variables do not have a value that will be populated in the engine (such as those that allow end users to import external pages manually) do not appear in the Outline Viewer panel in LiveEditor until end users add them to the Live document. However, if those same placeholder documents contain variables that will not be populated in the engine but are driven by sections, then they will

appear in the object tree as placeholder documents containing a single blank page. End users can replace these pages or append additional pages to these placeholder documents in LiveEditor.

- Placeholder documents that contain section data cannot be deleted using the Outline Viewer panel in LiveEditor.

For information about including placeholder documents in a Live document, see “[Creating a Placeholder Document](#)” on page 368.

10.3 Live Feature Support with Section Data

Because section data in a Live document goes through a different life cycle from section data used in non-interactive applications, some features are not supported when you use section data with Exstream Live. As a result, keep the following in mind as you work with section data:

- Section data in reference files is not supported.
- If you have licensed the Advanced Tables module, then you will be able to use the following table types to support section data in Exstream Live:
 - Automated tables with levels
 - Automated tables with sections
 - User tables
- Document combination features are not supported

For more information about section data and document combination in Exstream Live, see “[Disabling Document Combination for Section-Driven Live Documents](#)” below.

10.3.1 Disabling Document Combination for Section-Driven Live Documents

Because the underlying section data determines the order in which section-driven documents appear in a Live document, Exstream Live does not support document combination features.

To specify the document properties that are required for section-driven documents in Exstream Live:

1. In Design Manager, from the Library, drag the document to the Property Panel.
2. Click the **Targeting** tab.
3. In the **Document inclusion method** area, from the **Section/Node document**

combination drop-down list, select **Always starts a new document**.

4. Make sure that the **Allow documents to be combined** check box is not selected.
5. From the Menu bar, select **File > Save**.

Chapter 11: Using XML Node Data with Live Documents

XML node data allows you to connect objects or content in your design to specific elements or XML nodes in the data structure of a schema model data file and to use the data structure to drive the inclusion of content based on whether the node exists in the customer data. It is a powerful feature that you can use to design section objects, documents, and tables once, and then reuse them as required based on data. For example, suppose that you are setting up monthly statements for a mobile phone company in a traditional Exstream Design and Production application. The data for an account that uses a family plan might contain groups of records for each device on the account. By using XML nodes, you can design a single table—complete with headers, footers, and call details—that repeats as needed on the statement for each device on the account.

The basic principles of using XML nodes with Exstream Live are the same as the principles for using XML nodes to drive content in traditional Exstream applications. However, there are some differences, both in the way that XML nodes are processed and in the types of features that are supported with XML nodes. In addition, you must consider how XML nodes affect the way end users' changes appear and behave.

This chapter discusses the following topics:

- “[How XML Node-Driven Content Affects Editing in Live Documents](#)” below
- “[Live Feature Support with XML Nodes](#)” on the next page

For general information about schema model data files and XML node principles, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

11.1 How XML Node-Driven Content Affects Editing in Live Documents

From LiveEditor, end users cannot add XML node-driven objects to a Live document; however, end users can edit objects that are driven by XML nodes. It is important to understand the scope of where end user edits and changes are applied for XML node-driven objects and content. When you set up XML nodes and the application, keep the following considerations in mind:

Consideration	Description
Deleting XML node-driven objects from LiveEditor	If an end user deletes an object that is driven by XML nodes, the underlying data is not lost. However, the object will no longer appear in LiveEditor.

Consideration	Description
Using functions with XML nodes	If a function is fired within an object that is driven by an XML node, then that function affects only other objects that are driven by the same XML node. For example, if a function is run to update text inside an XML node-driven document, then only text in other documents that are driven by the same XML node is affected.
Using rules with XML nodes	Rules are run in the context of the XML node that you reference in the rule logic. Specifically, rules use the variable values within the scope of the XML node that you reference. The scope of the XML node can include the children of the XML node, key references, or the nearest relative of the XML node.
Using variables with XML nodes	The default number of elements in an array variable is zero. Therefore, a growing array variable will be empty until elements are assigned to it. If the array variable remains empty, it will not contain any editable content, and it will not drive automated content, such as table rows or paragraphs.
Using paragraph objects with XML nodes	For paragraph objects, if end users make edits to an interactive area, those edits are also applied to any other instance of the same paragraph object (regardless of whether that paragraph object is included in a section object that is driven by XML nodes), because the paragraph object is a single design object that is used in multiple locations. However, if you want to allow end users to make changes to a paragraph object that are unique to other instances of the same object, then you must use the following settings on the paragraph object: <ol style="list-style-type: none">1. You must include the paragraph object in an XML node-driven section object.2. In the schema model data file, you must map a variable as a child or descendant of the same XML node that drives the inclusion of the section object.3. In the paragraph object, include a reference to the variable that you added to the data file. End users can then edit the variable, and their changes to that variable would affect only that single instance of the paragraph object in the XML node-driven section object.
Using XML node-driven documents	Because the underlying data determines the order in which XML node-driven documents appear in a Live document, XML node-driven documents cannot be moved using the Outline Viewer panel in LiveEditor. For information about allowing end users to reorder documents in the Outline Viewer in LiveEditor, see <i>Designing for LiveEditor</i> in the Exstream Design and Production documentation.

11.2 Live Feature Support with XML Nodes

Because XML nodes in a Live document go through a different life cycle than XML nodes that are used in non-interactive applications, some features are not supported when you use XML nodes with Exstream Live. As a result, keep the following in mind as you work with XML nodes:

- If you have licensed the Advanced Tables module, then you will be able to use the following table types to support section data in Exstream Live:
 - Automated tables with levels
 - Automated tables with sections
 - User tables
- Document combination features are not supported

For more information about XML nodes and document combination in Exstream Live, see [“Disabling Document Combination for XML Node-Driven Live Documents” below](#).

11.2.1 Disabling Document Combination for XML Node-Driven Live Documents

Because the underlying XML node data determines the order in which XML node-driven documents appear in a Live document, Exstream Live does not support document combination features.

To specify the document properties that are required for XML node-driven documents in Exstream Live:

1. In Design Manager, from the Library, drag the document to the Property Panel.
2. Click the **Targeting** tab.
3. In the **Document inclusion method** area, from the **Section/Node document combination** drop-down list, select **Always starts a new document**.
4. Make sure that the **Allow documents to be combined** check box is not selected.
5. From the Menu bar, select **File > Save**.

Chapter 12: Managing Messaging and Marketing Content in Live Documents

Exstream Live offers advanced message design and management to let you build and integrate personalized marketing and informational messages into your Live documents. Campaigns and messages behave fundamentally the same in Live as they do in Exstream Design and Production, but there are two major differences. First, you can allow end users to select from a list of qualified campaigns for each customer in LiveEditor. Second, you can use interactive frames that allow end users to select from multiple messages. Both of these features give you more design flexibility by allowing end users to further target Live documents for specific customers.

For example, suppose you are designing a Live document for insurance agents. You can create marketing campaigns that showcase upgrades available with specific insurance products, such as earthquake or flood insurance, as part of a homeowners policy. You can then configure the Live document so that agents can choose from multiple messages that highlight the importance of the added insurance, based on a policyholder's geographical location.

When working with campaigns and messaging, keep in mind that the following are not supported in LiveEditor:

- Growing and shrinking graphic messages
- Background colors on graphic messages
- Interactive areas within graphic messages
- Text messages in interactive frames
- Multiple messages in interactive frames
- Teaser messages

Teaser messages are generally short messages that invite a customer to find out more information about the other messages in a campaign. Although end users cannot add teaser messages from within LiveEditor, you can add them to your fulfillment application so that they will appear after end users submit Live documents for fulfillment.

You can use document messages and message-driven pages in Live documents, provided that they qualify for inclusion based on the settings you specify in Design Manager.

For more information about setting up campaigns and marketing messaging in Design Manager, see *Managing Marketing Messages* in the Exstream Design and Production documentation.

This chapter discusses the following topics:

- “[Using Campaigns in Live Output](#)” on the next page
- “[Setting a Campaign Priority](#)” on the next page

- “Controlling How Graphic Messages Are Placed in a Message Frame” on the next page
- “Controlling How Messages Are Placed During Fulfillment” on page 211
- “Adding Messaging and Marketing Content in LiveEditor: The End User Experience” on page 211

12.1 Using Campaigns in Live Output

After you have created a campaign, complete the following steps to configure the campaign so that it is available for a Live application:

1. In Design Manager, from the Library, drag the campaign to the Property Panel.
2. Click the **Interactive** tab.
3. Select the **Allow Campaign to be used in Live output** check box.
4. From the Menu bar, select **File > Save**.

For more information about setting up and configuring campaigns and marketing messaging in Design Manager, see *Managing Marketing Messages* in the Exstream Design and Production documentation.

12.2 Setting a Campaign Priority

You can assign priorities to campaigns to specify how the engine will process them during fulfillment. For example, suppose you include campaigns as part of your fulfillment application, but you do not want them to replace campaigns placed by end users in LiveEditor. In that case, you can specify DLF campaigns as the top campaign priority so that messages that end users place in LiveEditor remain intact during fulfillment.

To give priority to campaigns placed by end users in LiveEditor:

1. In Design Manager, from the Library, drag the application to the Property Panel.
2. Click the **Marketing** tab.
3. From the **First campaign priority key** drop-down list, select **DLF campaigns**.
4. From the Menu bar, select **File > Save**.

For more information about specifying campaign priority in Design Manager, see *Managing Marketing Messages* in the Exstream Design and Production documentation.

12.3 Controlling How Graphic Messages Are Placed in a Message Frame

With message frames, you can specify whether or not end users are required to select a message, and you can specify whether a particular message is to be used in fulfillment.

This task assumes you have already created and placed the frame into your design. It is not necessary for you to create messages before setting up the frames; however, you must have already created any message types or message templates you intend to use.

For more information about creating and placing frames in your design, see *Designing Customer Communications* in the Exstream Design and Production documentation.

To control how messages are placed in a message frame:

1. Open the design page in Designer.
2. Double-click the frame.

The **Frame Properties** dialog box opens.

3. Click the **Interactive** tab.
4. From the **Content change** drop-down list, select one of the following options:
 - **Change is optional**—Select this option if end users are not required to select a message.
 - **Change is required**—Select this option if end users are required to select a message.
5. If you want the selected message to be retained if the DLF file is used as part of the fulfillment application, select the **Selected message must be used in fulfillment** check box.

During fulfillment, this setting will prevent a message from a lower-priority campaign from replacing the message that an end user places in this frame.
6. Click **OK**.
7. From the Menu bar, select **Edit > Save**.

12.4 Controlling How Messages Are Placed During Fulfillment

You can also use your fulfillment application to update or replace the campaigns used in a Live document. For example, suppose you distribute a Live document that allows end users to select from three campaigns: Campaign A, Campaign B, and Campaign C. However, after you distribute the Live document, you update Campaign A. Instead of having to repackage and redistribute the Live document, you can simply add the DLFCAMPDISABLE and PROMOTECAMPTODLF engine switches to the control file in the fulfillment application. The DLFCAMPDISABLE switch disables the existing campaign, and the PROMOTECAMPTODLF switch replaces it with the updated campaign. After the end user submits the Live document for fulfillment, the engine places the correct version of Campaign A in the final output.

The process is the same if you want to completely replace a campaign (for example, replacing Campaign E with Campaign F). In the control file for your fulfillment application, you can use DLFCAMPDISABLE to disable Campaign E, and then use PROMOTECAMPTODLF to replace it with Campaign F.

For more information about the DLFCAMPDISABLE and PROMOTECAMPTODLF engine switches, see [“LiveEditor and Live Engine Switches” on page 432](#).

When an end user places a message in a Live document and then submits it for processing, the engine determines which message the end user selected and updates the tracking database accordingly. Likewise, the tracking database also records when the fulfillment application replaces existing messages.

12.5 Adding Messaging and Marketing Content in LiveEditor: The End User Experience

For end users working in LiveEditor, the process of placing a marketing message in a frame is similar to the process of using an image selector to place an image in a frame.

To place a marketing message in a Live document, end users must complete the following steps:

1. In LiveEditor, the end user navigates to a customer document that uses messaging and marketing content.
2. The end user can use the **Campaigns** palette (available by selecting **View > Campaigns** from the Menu bar) to see a list of all campaigns that are available for the customer. Each

campaign in the list is accompanied by a check box that end users can select or clear so that messages from the campaign are shown or hidden, respectively.

3. The end user then navigates to an interactive message frame and double-clicks the frame to bring up the message picker dialog box.
4. The end user then scrolls through the thumbnail images available in the message picker dialog box and selects a message to place in the message frame.

Chapter 13: Using Barcodes in Live Documents

If you have licensed the High-Volume Delivery module, you can include barcodes in your Live documents to capture dynamic information that changes due to an end user's selections or changes. For example, you can include a barcode that is updated with the new number of total pages after the end user has finished making selections on an interview page. When the Live document is sent for fulfillment, the barcode can drive other processes based on the new size of the file. Or, you can print the Live document from LiveEditor and scan the barcode at the end user site to track the document.

Defining barcodes for Exstream Live is similar to defining barcodes for use in a traditional application, with the exception that not all barcode types and properties are supported in Live. Also, in Live applications, barcodes cannot be embedded in paragraph objects or tables.

For general information about creating and using barcodes in Exstream, see *Creating Output* in the Exstream Design and Production documentation.

This chapter discusses the following topics:

- “[Types of Barcodes Supported in Live Documents](#)” below
- “[Special Considerations for Generating Barcodes for Live Documents](#)” on the next page
- “[Special Considerations for Assigning Variables for Barcode Data in Live](#)” on page 215

13.1 Types of Barcodes Supported in Live Documents

You can use the following barcode types in a Live document:

- 3 of 9
- Code 128
- DataMatrix 2D
- EAN 128
- Four State
- General purpose
- Interleaved 2 of 5

- Modified Plessey
- PDF 417
- POSTNET
- UPC

The following barcode types are not supported in Live documents:

- GBR OMR
- Japanese Postal
- OMR
- QR Code

13.2 Special Considerations for Generating Barcodes for Live Documents

As with traditional Exstream Design and Production applications, you can use one of three methods to generate barcodes on a page:

- Generate a barcode with a specified font.
- Generate a barcode by drawing the lines.
- Import a scanned barcode to use in Designer.

For more information about specifying how a barcode is generated, see *Creating Output* in the Exstream Design and Production documentation.

Because generating a barcode by drawing the lines does not require fonts that end users might not have, you should choose to generate a barcode by drawing the lines when possible.

You can generate the following types of barcodes by drawing the lines:

- 3 of 9
- 2 of 5
- Code 128
- Modified Plessey
- UPC

If you must use a barcode that can only be drawn with fonts, you must make sure end users have the proper fonts set up on their computers, or you must include the proper fonts in the DLF

file. If an end user does not have the required barcode font, the operating system uses the font that most closely matches the font used to display the barcode, which might be incorrect for a barcode font. The end user receives a warning when printing or saving in PDF format a Live document that uses substituted fonts.

For information about including fonts in a DLF file, see [“Controlling How Fonts are Handled When Packaging a DLF File” on page 251](#).

13.3 Special Considerations for Assigning Variables for Barcode Data in Live

Keep in mind that the barcode data is updated at the variable substitution time you select from the **When to substitute** drop-down list in the properties of each variable used in the barcode. (Or, if you select **Use Settings default**, the variable substitution time you select from the **Default variable substitution time & rule execution time** drop-down list is used.) For example, if you select **Initial Engine and Interactive Editor** as the variable substitution time, the barcode is updated during the first engine run and when values change in LiveEditor, but not during fulfillment in the engine. Make sure you select a variable substitution time that updates the barcode data as needed to support your workflow. For example, suppose that an end user prints a Live document locally, scans the barcode, and then sends the Live document to customers. In this case, the variable substitution time must allow the barcode variables to be updated in LiveEditor so the barcode reflects the end user's changes in LiveEditor.

For more information about variable substitution time, see [“Setting the Default Variable Substitution and Rule Execution Time” on page 351](#).

Also keep in mind that page numbering variables have unique behaviors in Exstream Live. These behaviors apply if, when assigning the barcode data on the **Contents** tab of the barcode object properties in Design Manager, you select a page numbering option from the **Content Type** drop-down list, or you select **Variable** and select a page numbering system variable in the **Value** variable box. The variables that determine the value of the page numbering options available in the **Content Type** drop-down list do not restart at the beginning of each document within the Live document; that is, they are based on consecutive page numbering for the entire set of documents for an individual customer. Also, barcodes that contain page numbering are updated based on the variable substitution times specified for the corresponding page numbering variables.

The value for each of the following page numbering options is determined by the listed page number variable:

- **Total Pages in Doc**—'SYS_PageTotalInDocument'
- **Page in Document**—'SYS_PageInDocument'
- **Page in Queue**—'SYS_PageInQueue'
- **Page in Stream**—'SYS_PageInBreak'

For information about using page numbering variables in Live, see “[Unique Behaviors of Page Numbering System Variables in Live Documents](#)” on page 429.

Chapter 14: Using Tools for Providing Guidance to End Users

An important role of the Live document designer is to make the Live documents as easy and self-explanatory to use as possible. Exstream Live offers design tools to help you do this; for example, selection controls and built-in intelligence can help make Live documents intuitive for end users to navigate and interact with. However, in addition to these features, Exstream Live also provides some features that are specifically designed to help you communicate with end users and provide guidance as they work with Live documents. You are not required to use these objects; rather, they are intended for you to use as needed in your environment.

You can use the following tools to provide guidance to end users:

- **Themes**—A collection of settings, such as highlighting or text formatting, that indicates to end users the types of changes that they must make in a Live document
- **Pop-up information**—Messages that appear over areas in Live documents that describe the object or how end users must interact with the object

This chapter discusses the following topics:

- “[Using Themes to Visually Guide End Users](#)” on the next page
- “[Providing Pop-Up Information to Describe Specific Objects](#)” on page 224

14.1 Using Themes to Visually Guide End Users

Themes are non-printing objects used in Exstream Live to provide visual guidance in documents in LiveEditor. Themes use a collection of settings such as text color, highlighting, and borders to indicate to end users the different types of changes that can be or must be made in a Live document. For example, in the following illustration, different colors indicate the types of changes that need to be made in the document, as well as the status of two hyperlinks. Specifically, the yellow outline around the image selector indicates that a change is required, while the light blue highlighting indicates the changes can optionally be made. The different colors applied to the hyperlink text indicate a hyperlink that has been clicked and a hyperlink that has not yet been clicked.

Example of theme

Dear **Customer name**

Diam graeci nec at, duo ad nibh ubique doctus. Paulo diceret eam id. Vis verear nostrum eu, alii movet reprehendunt eam no. Ex vim aliquip delectus assueverit, usu suas scriptorem at, mea scaevola appellantur no.

Dicta omnium repudiandae ex cum. Ius at vide invidunt perfecto, in nec facilis interesseret. Et noluisse gloriatur has, duo eu mazim epicurei vulput.

www.hyperlink.com

www.hyperlink.com



Note: The example uses external hyperlinks. You cannot apply a theme to internal hyperlinks.

Because themes are non-printing, they appear only in LiveEditor. When the Live document is processed or printed from LiveEditor, the theme is not visible.

As a Live document designer, you can create one or more themes to be used in Live documents. You can customize the settings of the themes so they are easy for your end users to understand. Themes can be as simple as using one color of text to indicate areas where end users must make a change, or as complex as using a variety of text colors and formatting to indicate specific requirements, such as objects that must be changed or hyperlinks that must be clicked. Additionally, you can create multiple themes that can be used for a single Live document, and allow end users to select the theme they want to use. For example, you might create one theme for front-end users and a different, simplified theme for final reviewers.

When you design a theme, keep in mind the following principles:

- If you allow only text editing on a text box, the text themes apply to all text in the text box. The object themes are not used.
- If you allow text editing on a text box (or have separate editable text areas) and other properties, such as the ability to move the text box, the text themes and the object frame themes are used. If the theme specifies a fill color, the object will not be filled, however.
- If you do not allow any text editing inside a text box (at the text box level or individual level), but allow other editability, the object fill theme is used. For example, you can have an address box that end users cannot edit but can move as needed for their envelopes.

Also, keep in mind that themes do not enforce specific editing actions; they are used only to indicate different types of interactive areas on a Live document.

To use themes to visually guide end users, complete the following tasks as needed:

- “[Creating and Setting Up Themes](#)” below
- “[Specifying How Themes are Used in LiveEditor](#)” on page 221
- “[Controlling How Themes are Applied to Selection Controls](#)” on page 222
- “[Previewing and Testing Themes](#)” on page 223

14.1.1 Creating and Setting Up Themes

1. In Design Manager, in the Library, right-click the **Themes** heading and select **New Theme**.

The **New Theme** dialog box opens.

2. In the **Name** box, enter a name. In the **Description** box, enter a description (optional).
3. Click **Finish**.

The theme opens in the Property Panel for you to define.

4. Use the **Group** box and **State** box to select a combination of objects and states for which you want to define a theme setting. The **Group** box contains a list of types of objects that can be interactive in a Live document and the **State** box contains a list of all the possible states each type of object can have in a Live document. For example, to define the theme setting for text that must be edited but has not yet been changed, select **Required text** from the **Group** box and **Active** from the **State** box.

The **Group** box contains the following options:

Group options

Option	Description
Text	Text and variables in text boxes and tables that may be or cannot be edited
Required text	Text and variables in text boxes and tables that must be edited
Text link	Hyperlink text
Object	Any object on a design page, including text boxes and tables, that may be or cannot be edited
Required object	Any object on a design page, including text boxes and tables, that must be edited
Object link	Any object on a design page that acts as a hyperlink
Interview page	Pages on which the Live interview page check box has been selected. If you select Interview page , no options are available in the State box; you can only control the background of an interview page. For information about interview pages, see " Using Interview Pages in a Live Document " on page 125 .
Recipient page	Pages which have been selected on the Recipient Data Page box or the Recipient Cover Page box on a recipient profile's properties. The theme you create for a recipient page applies to all of the pages associated with a recipient: the recipient data page, the cover page, and the pages that make up the recipient copy. For information about recipient pages, see " Allowing End Users to Add and Modify Copies of Documents " on page 178 .

The **State** box contains the following options:

State options

Option	Description
Active	The text or object is in the process of being edited.
Already visited	The hyperlink has been followed.
Changed	The text or object has been edited.
Deleted	The text or object has been removed.
Disabled	The text or object is editable but the user does not have access to it.
Hover	The text or object is editable but the user does not have access to it.

State options, continued

Option	Description
Invalid	The value entered does not meet the specified requirements (for example, a telephone number is entered into an area that expects a name). This state overrides any other states that apply to the text at the same time.
Uneditable	The text or object cannot be edited.
Untouched	The text or object can be edited but has not been, or the hyperlink can be followed but has not been.

5. After selecting a combination of options from the **Group** box and **State** box, select the **Active** check box if you want the settings you specify for that combination to appear in LiveEditor. After defining the settings, you can clear the **Active** check box if you want to retain the settings but do not want them to be currently used in the theme.
6. After selecting a combination of options from the **Group** box and **State** box, use the other options in the Property Panel to specify the appearance of the object in the specified state. Not all formatting options are available for all combinations of states and groups. You can use the **Sample** area to see an example of how the combinations you choose will appear in LiveEditor.
7. From the Menu bar, select **File > Save**.

14.1.2 Specifying How Themes are Used in LiveEditor

After defining a theme, you must associate it with a setting so that it is included in the Live document and is available in LiveEditor.

You can also associate multiple themes with a Live document in order to allow different end users to use different themes when they edit the Live document. For example, if some users are accustomed to seeing text areas that must be edited in red but other users are accustomed to seeing text areas that must be edited in blue, you can create two different themes that are designed for both requirements. Then, on the setting object, you associate both themes with the Live document. In LiveEditor, end users can choose which theme to use on the **Live Document Options** dialog box, which is accessed by selecting **Tools > Options** from the Menu bar.

Before specifying how themes can be used in LiveEditor, you must have created one or more themes.

To specify how themes are used in LiveEditor:

1. In Design Manager, from the Library, drag the setting object to the Property Panel.
2. Click the **Editor Framework** tab.

3. Use the **Available themes** box to specify one or more themes that will be used with the Live document:

- a. Below the **Available themes** box, click .

The **Select Theme** dialog box opens.

- b. Select a theme you want to be available in LiveEditor and click **OK**.

The **Select Theme** dialog box closes and the theme you selected appears in the **Available themes** box.

- c. If you want to include multiple themes so that end users can switch between themes in LiveEditor, repeat step a through step b to select all the themes you want to be available in LiveEditor. Otherwise, skip to step 4 to select the theme that is used by default in LiveEditor.

4. In the **Default theme** box, click .

The **Select Theme** dialog box opens.

5. Select the theme that you want to be used by default in LiveEditor and click **OK**.

The **Select Theme** dialog box closes and the theme you selected appears in the **Default theme** box.

6. From the Menu bar, select **File > Save**.

14.1.3 Controlling How Themes are Applied to Selection Controls

By default, check boxes, radio buttons, and the check boxes or radio buttons associated with a selection control are not affected by the theme used in the Live document. However, you can specify that these objects use the theme that is in effect for the rest of the document to indicate the state of the object (for example, active, changed, and so on).

When you apply themes to selection controls, the settings defined for the object group (for example, radio buttons) are applied to the selection controls.

To control how themes are applied to selection controls:

1. In Design Manager, from the Library, drag the setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. Select the **Use object active theme on activation buttons** check box.
4. From the Menu bar, select **File > Save**.

14.1.4 Previewing and Testing Themes

As you set up the properties of a theme, you can use Designer to see a preview of how the theme will appear to end users in LiveEditor. For example, you can make sure that you have designed enough contrast between different states in the theme or that the theme makes it easy for end users to see all the areas that require change. When you view a theme in Designer, you use the Live Preview mode of Designer.

To preview and test a theme:

1. In Designer, open any page that contains the type of component with which you want to test the theme. For example, to test the theme conventions for required text, open a page that contains text that end users must change.
2. From the Menu bar, select **Tools > Options**.

The **Designer Options** dialog box opens.

3. Click the **Designer** tab.
4. In the **Exstream Live Simulation Mode** area, specify the theme you want to use in Live Preview.

- a. Click .

The **Select Theme** dialog box opens.

- b. In the box, select the theme you want to use.
- c. Click **OK**.

The **Select Theme** dialog box closes and the theme you selected appears in the **Exstream Live Simulation Mode** box.

5. If you want a watermark to appear in Live Preview to help you distinguish between Live Preview mode and design mode, select the **Show Live watermark in Live mode** check box.

6. Click **OK**.

The **Designer Options** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

8. On the Properties toolbar, click .

Live Preview opens.

You can interact with and test the Live document as needed in Live Preview mode. Click  to toggle between Live Preview mode and the design mode.

14.2 Providing Pop-Up Information to Describe Specific Objects

You can use pop-up messages to provide detailed information about specific objects in a Live document. For example, the pop-up feature is often used to provide tips to end users about how they should interact with a specific object, examples of the type of changes to make, or detailed descriptions of the object. You provide this information in the **Tip and description** box on the **Interactive** tab of the object properties.

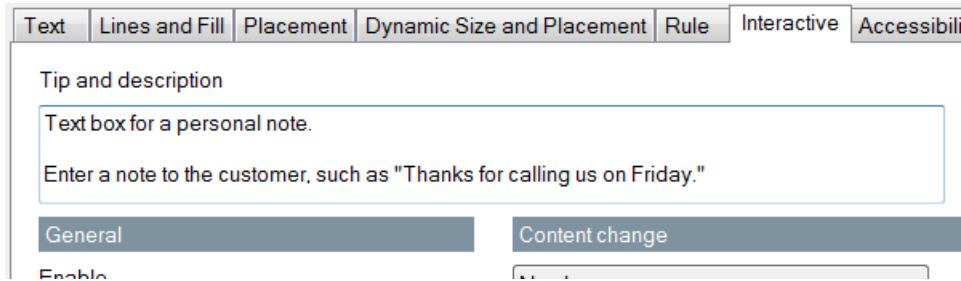
The information you enter in the **Tip and description** box is used in several ways in LiveEditor. The first line you enter appears as a title in the pop-up message that appears when end users

click [?] on the Live Actions toolbar or when users navigate to the next or previous interactive area using the menu or toolbar. This information also appears when end users hover over the object, so it is ideal for providing a descriptive name for the object. The second line (or any subsequent lines) you enter in the **Tip and description** box appears only in the pop-up

message that opens when end users click [?], so it is useful for providing more detailed instructions or examples of how to interact with the object.

The following examples illustrate how pop-up information appears in Designer when you create it and how it appears to end users in LiveEditor.

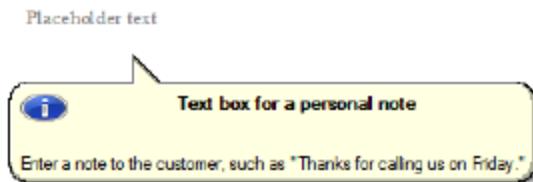
Appearance of tip and description information in Designer



Appearance when end users hover over object



Appearance when end users click the interactive area help button or navigate to interactive areas



Tip: The information provided in the **Tip and description** box is used differently from the information in the **Live caption** box. Descriptive text that appears in the **Live caption** box is used to describe objects in the Outline Viewer.

For more information about customizing the Outline Viewer appearance, see “[Customizing the Outline Viewer](#)” on page 243.

To provide pop-up information about objects:

1. Open the properties of the object for which you want to provide pop-up information.
 - To open the properties of a section or paragraph in the Library, drag the object to the Property Panel. (You cannot provide pop-up information for pages and documents.)
 - To open the object of a design object, such as a text box, select the object and click .
2. Click the **Interactive** tab.
3. In the **Tip and Description** box, enter the pop-up information for the object.
 - a. In the first line of text, enter the information you want to appear when end users hover over the object and as the title when end users click .
 - b. In the second line (or subsequent lines) of text, enter the information you want to appear only when end users click .
4. From the Menu bar, select **Edit > Save**.

Tip: You can use the **LiveBalloon** and **LiveBeep** built-in functions to customize the appearance and behavior of the pop-up message that appears when end users click .

For information about Live built-in functions, see “[Live Built-In Functions](#)” on page 381.

Chapter 15: Customizing the End User Interface and Navigation Options

You can make it easier and more intuitive for end users to work in particular Live documents by tailoring the access to tools and the navigation in LiveEditor for each document. You can control the commands, toolbars, and panels that are available to end users, and you can also create custom toolbars that provide quick access to commands that are commonly used for an individual Live document. You can also help end users quickly navigate the Live document in a variety of ways.

This chapter discusses the following topics:

- “[Customizing the End User Interface](#)” below
- “[Customizing Navigation](#)” on page 235

15.1 Customizing the End User Interface

To make it more intuitive for end users to work with Live documents in LiveEditor, you can customize the LiveEditor interface for each Live document to include only the tools end users need for that particular Live document. You can select which panels, toolbars, commands, and custom commands are available for each Live document. For example, suppose you do not include any tables or text that needs formatting in your Live document. In this case, you might remove the Formatting toolbar and the **Table** menu to simplify the interface. Additionally, if there are particular commands or actions that end users commonly use for a particular Live document, you can add them to a custom toolbar for easy access.

This section discusses the following topics:

- “[Using Views to Customize the End-User Interface](#)” below
- “[Setting Up a Custom Toolbar to Provide End Users with Quick Access to Common Tasks](#)” on page 233

15.1.1 Using Views to Customize the End-User Interface

To specify which panels, toolbars, commands, and custom commands are available in a Live document, you must set up a view object. You can make multiple views available for a Live document. For example, you might include a default "Basic" view that includes only the basic tools an end user needs to use for your Live document, such as general editing tasks, text

formatting, and navigation, and you might include an additional "Advanced" view that includes advanced tools, such as revision tools and the Debugger panel.

Note: To specify which commands and interface options are available with a view, you must have LiveEditor installed on your computer.

To use a view, you must complete the following tasks:

1. ["Setting Up a View to Remove or Add Options From the End-User Interface" below](#)
2. ["Associating Views With a Live Document" on page 232](#)

Setting Up a View to Remove or Add Options From the End-User Interface

1. In Design Manager, in the Library, right-click the **Views** heading and select **New View**.
The **New View** dialog box opens.
2. In the **Name** box, enter a name. In the **Description** box, enter a description (optional).
3. Click **Finish**.
The new view opens in the Property Panel for you to define.
4. Click the **Editor Framework** tab.
5. To remove or add options from the view, complete the following tasks as needed:

To	Do this
Select which built-in commands are available in LiveEditor	Complete the steps described in "Selecting Which Built-in Commands Are Available in LiveEditor" below .
Select custom actions to make available in LiveEditor	Complete the steps described in "Selecting Custom Actions to Make Available in LiveEditor" on page 229 .
Select which toolbars, panels, and associated groups of commands are available in LiveEditor, and whether rulers and the status bar are shown	Complete the steps described in "Selecting Which Toolbars and Panels Are Available in LiveEditor" on page 230 .
Select a custom toolbar to appear in LiveEditor	Complete the steps described in "Selecting a Custom Toolbar to Appear in LiveEditor" on page 231 .

6. From the Menu bar, select **File > Save**.

Selecting Which Built-in Commands Are Available in LiveEditor

To select which built-in commands are available in LiveEditor, you use the **Commands to display** list in the view object properties. When you clear the check box for a command, it does

not appear on the menu in LiveEditor, it is disabled on the toolbar where it appears, and the end user cannot activate the associated function using the keyboard shortcut. For example, if you do not want end users to be able to print, clear the **Print** check box.

By default, all menu commands are available.

Keep in mind that to specify which commands and interface options are available with a view, you must have LiveEditor installed on your computer.

Also keep in mind that certain built-in commands depend on properties of the setting object selected for the Live document. Therefore, some commands that are available in the view may not appear in LiveEditor because you have not configured the associated properties in the setting object. For example, in order for end users to use the **Store** and **Submit** commands, you must use the **Functions to execute on LiveEditor events** box to specify functions associated with those commands. Also, the **Upload distribution list file** command is available only if you select the **Live document is a template** check box. For more information about setting up a specific command in the settings object, see the information for the specific command.

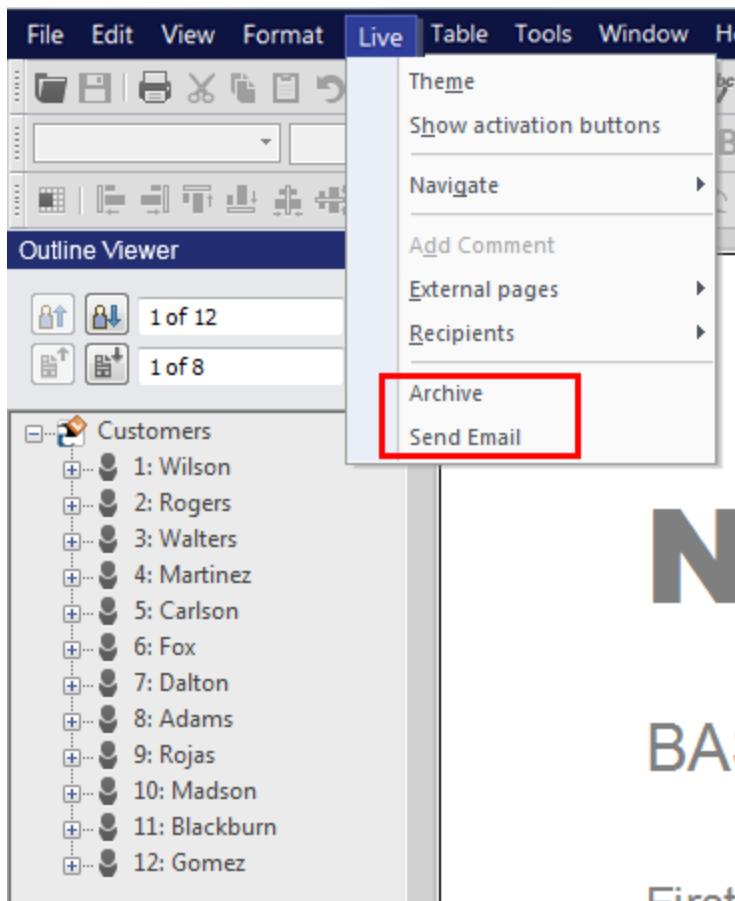
To select which built-in commands are available in LiveEditor:

1. In Design Manager, from the Library, drag a view object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Commands to display** box, select the check box for each command you want to be available in LiveEditor, or clear the check box for each command you do not want to be available in LiveEditor.
4. From the Menu bar, select **File > Save**.

Selecting Custom Actions to Make Available in LiveEditor

You use the **Custom commands** list in the view object properties to specify which Live actions are available for a Live document. Actions you add to the list are available from the **Live** menu in LiveEditor. You can also add them to a custom toolbar if you include one.

Actions on the Live menu in LiveEditor



For information about setting up actions, see “[Using Actions to Automate Editing Tasks](#)” on page 129.

For more information about setting up a custom toolbar, see “[Setting Up a Custom Toolbar to Provide End Users with Quick Access to Common Tasks](#)” on page 233.

To select which custom actions are available in LiveEditor:

1. In Design Manager, from the Library, drag a view object associated with the Live document to the Property Panel.
2. Click the **Editor Framework** tab.

3. Under the **Custom commands** list, click .

The **Select Action** dialog box opens.

4. Select an action from the list and click **OK**.

The selected action appears in the **Custom commands** list and will appear on the **Live** menu in LiveEditor.

5. Repeat step 3 through step 4 to add additional actions. You can add up to 50 actions.
6. If you want to change the order of an action on the **Live** menu, select the action in the **Custom commands** list and click  or  as needed. If you want to remove an action from the list, select it from the list and click .
7. From the Menu bar, select **File > Save**.

Selecting Which Toolbars and Panels Are Available in LiveEditor

You use the **Toolbars to display** box in the view object properties to specify which toolbars and panels are available, and whether the rulers and status bar are displayed. By default, all toolbars, panels, rulers, and the status bar are available.

The end user can select whether to show or hide available toolbars, panels, rulers, and the status bar within LiveEditor. Commands from toolbars hidden by the end user are still available in the LiveEditor menus.

To select which toolbars and panels are available in LiveEditor:

1. In Design Manager, from the Library, drag a view object associated with the Live document to the Property Panel.
2. Click the **Editor Framework** tab.

3. To select which items are available in LiveEditor, use the check boxes in the **Toolbars to display** box:

To	Do this
Select which toolbars are available	<p>Select or clear the following check boxes as necessary:</p> <ul style="list-style-type: none">• Standard—Used to complete general editing tasks• Formatting—Used to format editable text• Live Actions—Used to perform the custom actions included in the Custom commands list• Navigation—Used to go to specific areas in the Live document• Properties—Used to change the frame properties of editable objects• Position—Used to modify the positioning and layout of design elements• Review—Used to control revision tracking• Recipients—Used to add, remove, and change recipient copies• Campaigns—Used to manage campaigns associated with the Live document <p>When you clear the check box for a toolbar, it is not available in LiveEditor, and the commands that appear on the toolbar are also removed from menus in LiveEditor and cannot be activated using keyboard shortcuts. For example, if you do not want end users to be able to use any functions related to recipient copies, clear the Recipients check box.</p>
Select which panels are available	<p>Select or clear the following check boxes as necessary:</p> <ul style="list-style-type: none">• Debugger—Used to view and test the logic in a Live document• Outline Viewer—Used to navigate through and view the hierarchy of the Live document• Design Layers Palette—Used to enable or disable the design layers included in the Live document
Specify whether the rulers that appear in the design window are available	Select or clear the Rulers check box.
Specify whether the status bar that displays general information about the Live document is available	Select or clear the Status Bar check box.

4. From the Menu bar, select **File > Save**.

Selecting a Custom Toolbar to Appear in LiveEditor

You can include a custom toolbar in a view to provide end users with easy access to certain commands.

Note: In order for an action that is included on the selected custom toolbar to be available in LiveEditor, it must be added to the **Custom commands** list. If you add a custom toolbar that includes actions not included in the view, you are prompted to add those actions to the view.

For more information about custom toolbars, see [“Setting Up a Custom Toolbar to Provide End Users with Quick Access to Common Tasks”](#) on the next page.

To add a custom toolbar to a view:

1. In Design Manager, from the Library, drag a view object associated with the Live document to the Property Panel.
2. Click the **Editor Framework** tab.
3. Click the **Custom Toolbar** box.
4. Select a custom toolbar. (If you do not want to include a custom toolbar, select **No Toolbar**.)
5. Click **OK**.
6. From the Menu bar, select **File > Save**.

Associating Views With a Live Document

To apply the options you specify in a view object to a Live document, you must add the view to the Live setting object associated with the Live document. You can make multiple views available for a Live document. For example, you might include a default "Basic" view that includes only the basic tools an end user needs to use for your Live document, such as general editing tasks, text formatting, and navigation, and you might include an additional "Advanced" view that includes advanced tools, such as revision tools and the Debugger panel. LiveEditor would use the default "Basic" view when the end user opens the Live document. The end user could then select between the "Basic" and "Advanced" views in the **Live Document Options** dialog box.

To associate views with a Live document:

1. In Design Manager, from the Library, drag the setting object associated with the Live document to the Property Panel.
2. Click the **Editor Framework** tab.
3. Under the **Available views** list, click  .
The **Select View** dialog box opens.
4. Select a view from the list and click **OK**.

The selected view appears in the **Available views** list and will be available in LiveEditor.

5. To add additional available views, repeat step 3 through step 4, as needed.
6. If you want to change the default view that is used when an end user opens the Live document in LiveEditor, click  in the **Default view** box and select a default view.
7. From the Menu bar, select **File > Save**.

For more information about using setting objects and creating Live output, see “[Setting Up the Objects Necessary for Live Output](#)” on page 354.

15.1.2 Setting Up a Custom Toolbar to Provide End Users with Quick Access to Common Tasks

You can create a custom toolbar to provide end users with easy access to the actions and standard commands used frequently for a particular Live document.

To set up a custom toolbar:

1. In Design Manager, in the Library, go to **Environment > Interactive > Toolbars**.
2. Right-click the **Toolbars** heading and select **New Toolbar**.
The **New Toolbar** dialog box opens.
3. In the **Name** box, enter a name. In the **Description** box, enter a description (optional).
4. Click **Finish**.

The toolbar opens in the Property Panel for you to define.

5. Use the options available to configure the commands on the toolbar:

To	Do this
Add a built-in command	<ol style="list-style-type: none">In the Built-in Commands list within the Available commands list, highlight the built-in command you want to add to the toolbar.If you want to add the new command after an existing command, highlight the existing command in the Toolbar content list.Click 
Add an action	<ol style="list-style-type: none">In the Custom Commands list within the Available commands list, highlight the action you want to add to the toolbar.If you want to add the new command after an existing command, highlight the existing command in the Toolbar content list.Click  <p>For information about creating actions, see "Using Actions to Automate Editing Tasks" on page 129. For information about using custom icons for actions in a toolbar, see "Specifying an Icon to Display in a Toolbar" on page 131</p>
Remove a command	<ol style="list-style-type: none">In the Toolbar content list, highlight the command you want to remove from the toolbar.Click 
Add a separator line between commands	<ol style="list-style-type: none">In the Available commands list, highlight Separator.In the Toolbar content area, highlight the command you want the separator to appear after.Click 
Reorder commands	<ol style="list-style-type: none">In the Toolbar content area, highlight the command you want to reorder.To move the command in the list, click  or 

6. From the Menu bar, select **File > Save**.

Keep in mind that in order to make a custom toolbar available in a Live document, you must specify the custom toolbar in a view that is in turn included in the setting object associated with the Live document.

For more information about specifying a custom toolbar in a view, see ["Selecting a Custom Toolbar to Appear in LiveEditor" on page 231](#).

For more information about adding a view to a setting object, see ["Associating Views With a Live Document" on page 232](#).

15.2 Customizing Navigation

You can customize navigation to help end users navigate through the document in a logical way.

This section discusses the following topics:

- “[How LiveEditor Handles Basic Navigation between Editable Areas](#)” below
- “[Enabling Navigation to Editable Areas](#)” on the next page
- “[Customizing the Order of Navigation Between Editable Areas](#)” on page 237
- “[Creating Additional Navigation Controls](#)” on page 241
- “[Allowing End Users to Navigate From the Table of Contents](#)” on page 241
- “[Adding Automatic Navigation to Specific Interactive Areas](#)” on page 242
- “[Customizing the Outline Viewer](#)” on page 243

15.2.1 How LiveEditor Handles Basic Navigation between Editable Areas

When you design a Live document that contains editable areas (also known as "interactive areas"), LiveEditor allows end users to navigate between those areas using either the toolbar or keyboard shortcuts. LiveEditor uses the **Tab stop** option that you configure in Designer to enable navigation between interactive areas. By default, **Tab stop** is enabled for editable areas like text boxes, and it controls how end users navigate when they use the toolbar buttons and the TAB key:

- **Navigation using the toolbar buttons**—To navigate through interactive areas using the mouse, an end user can click  and  from the Navigation toolbar in LiveEditor.
- **Navigation using the TAB key**—To navigate through interactive areas using the keyboard, an end user can press the TAB key, just as you would do in a web form.

Tip: One important thing to remember is that if end users press the TAB key while they are making changes to text in an interactive area, LiveEditor will add a tab space to the text they are editing. To remove focus from the selected interactive area and move focus to the next interactive area, end users must press the CTRL+TAB keys on their keyboards. If you want end users to use only the TAB key to navigate between editable areas, use form fields or data fields (for PDF forms) instead of interactive areas in your Live document.

For information about setting up form fields or data fields, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

Suppose, for example, that a CSR must enter basic personal information for a customer who is opening a new account. After completing the **First Name** field, the CSR can press TAB or click to move the cursor directly to the **Last Name** field.

Using either of these methods takes end users only to the areas of the Live document that they can change. When using the Live navigation buttons, the cursor moves between interactive areas across the entire document, even if it contains multiple customers. When using the TAB key, however, the cursor stops at the last page in a customer, and the end user must manually open the next customer's pages or use the toolbar buttons to proceed.

By default, the tab order is determined by the order in which the Live properties were set for each object. For example, if you create a text box first on a page, and then add an image, if you set the Live properties for the image before you set the Live properties for the text box, then end users will navigate to the image first using either the Live navigation buttons or the TAB key. To view the order of navigation (known as "tab order" in LiveEditor), click on the Interactive Design toolbar.

15.2.2 Enabling Navigation to Editable Areas

You use the **Tab stop** check box on the **Interactive** tab of the object properties to let end users navigate to the object or text area in LiveEditor. When you select this option, the object is assigned a tab index number that indicates the order in which the end user navigates to the object or area when pressing the TAB key or clicking from the Navigation toolbar. To view the order of navigation (known as "tab order" in LiveEditor), click on the Interactive Design toolbar.

For information about changing the order of navigation, see "[Customizing the Order of Navigation Between Editable Areas](#)" on the next page.

The **Tab stop** option is enabled by default for objects that are typically used for editable areas, such as text boxes.

To enable navigation to an editable area:

1. In Designer, select an object to which you want to enable navigation and click .

The object properties open.

2. Click the **Interactive** tab.
3. Select the **Tab stop** check box.
4. Click **OK**.
5. From the Menu bar, select **Edit > Save**.

15.2.3 Customizing the Order of Navigation Between Editable Areas

By default, the order in which the cursor goes to editable areas (known as "tab order" in LiveEditor) is determined by the order in which you set Live properties for them. Because you do not always design objects in the same order in which end users need to edit them, you can customize the order of navigation. For example, if you set up the **Address Line 2** text box before the **Address Line 1** text box, you might want to change the order so the cursor goes to the **Address Line 1** text box first.

When customizing the order of navigation between editable areas, keep in mind the following behaviors:

- You cannot set the navigation order of objects on a page template.
- If an interactive variable is located in the middle of a larger editable area of text (for example, an editable paragraph), the variable is skipped when navigating through the area. For example, the first time the end user presses TAB or clicks , the cursor is placed at the beginning of the editable paragraph. Each time the end user clicks , the cursor is placed in the next editable area, not the variable inside the editable paragraph. If the end user places the cursor within the variable, the movement is restricted to the variable.
- You cannot set up a custom navigation order for objects contained by another object. For example, suppose an end user clicks  to navigate to an editable table. Each time the end user clicks , the cursor moves to individual editable areas within the table, such as cells or paragraphs, in left-to-right order. You cannot change this order.
- You can design a navigation order for the current page only. In other words, you cannot make an object on Page 4 the next object when an end user presses TAB in an object on page 3.
- If your design uses tab-ordered library components, Designer honors the tab order that was set when the library component was created. For information about using tab-ordered library components in your design, see ["How Designer Handles Library Components that Specify Tab Order" on the next page](#).

To change the navigation order for editable areas:

1. In Design Manager, from the Library, drag the page on which you want to customize the navigation order to the Edit Panel.

The page opens in Designer. This page should already contain the objects for which you want to set the navigation order.

2. On the Interactive Design toolbar, click  .

The navigation order of each object appears in a box next to the object.

Sample navigation order

Please complete the form below:

 1	First name
 2	Last name
 3	Address

3. Select an object for which you want to change the order of navigation. For example, if you want to make the object currently labeled as "3" in the navigation order be the first object in the navigation order, select object 3.
4. On the Interactive Design toolbar, click  to move the object up in the navigation order, or click  to move the object down in the navigation order.

The numbers shown beside the objects on the page change to reflect the new navigation order.

5. Complete step 3 through step 4 to customize the order for as many objects as you want to on the page.
6. From the Menu bar, select **File > Save**.

How Designer Handles Library Components that Specify Tab Order

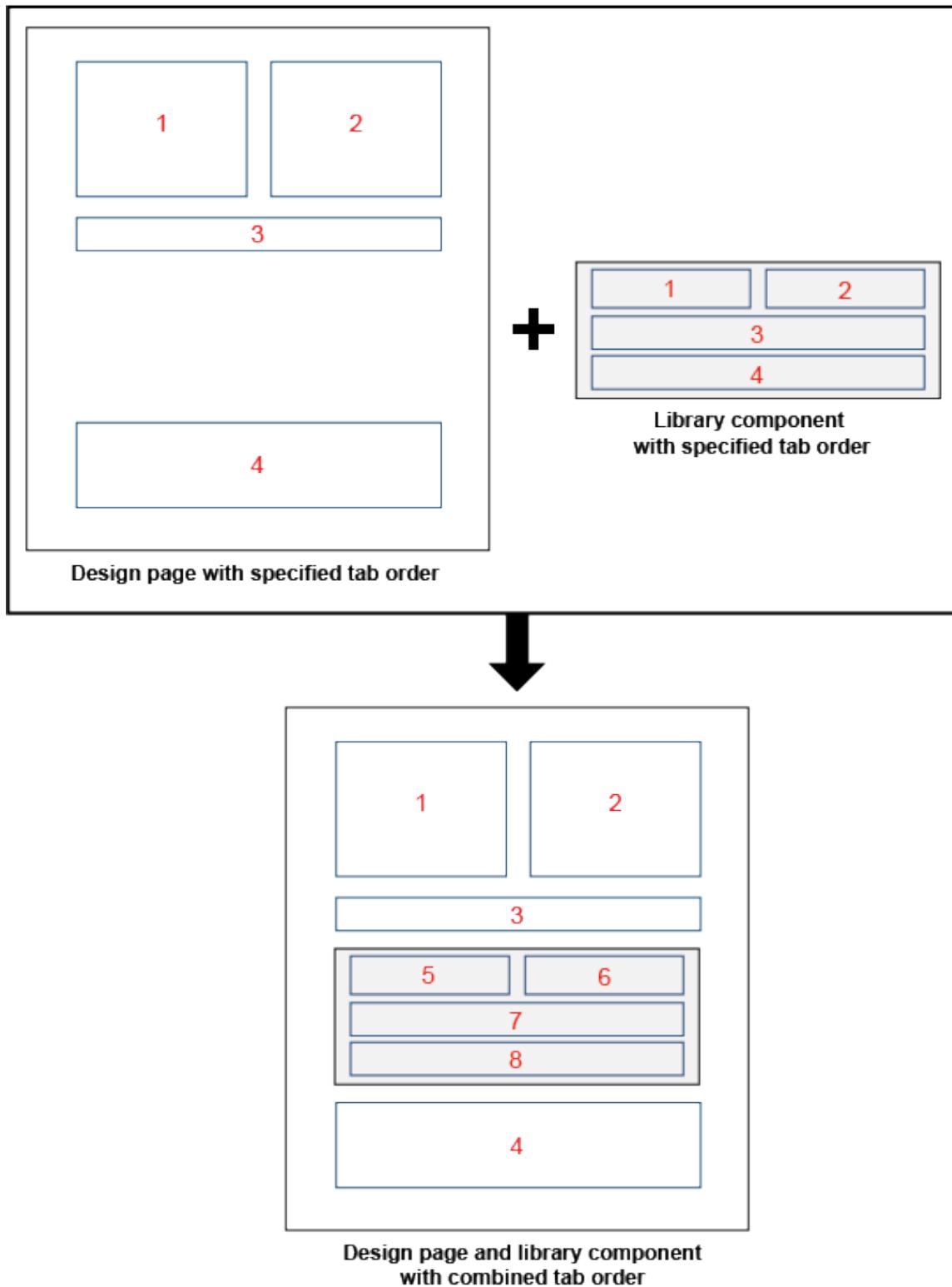
In some cases, you might want to add a library component that uses tab navigation to your design. When you add a tab-ordered library component to your design, Designer maintains the tab stops in that component as a group. For example, suppose that your design contains three editable components with a tab order of [1,2,3]. Now, suppose that you have an editable library component with a tab order of {1,2,3}. When you add the library component to your design, Designer automatically begins the numbering for the component after the numbering for any tab

stops that are already in your design. This means that if you add the library component after the second tab stop in your design, the overall tab order becomes [1,2,{4,5,6},3].

In that scenario, you can still use the or buttons to rearrange the tab order in your design, but Designer always moves the tab stops for the library component as a group—for example, [1,{2,3,4},5,6]. In order to change the tab order of the elements within a library component, you must unlock or open the library component separately in Designer.

The following figure further illustrates how Designer handles a mixture of library and non-library components that specify tab order:

Combined tab order for a design page and a library component



For information about adding library components to your design, see *Designing Customer Communications* in the Exstream Design and Production documentation.

15.2.4 Creating Additional Navigation Controls

You can customize the way end users navigate in a Live document by adding additional controls for basic navigation to general areas of a Live document, such as navigation to a document or page relative to the current cursor position. For example, you might add buttons to the bottom of a page to allow end users to go to the next or previous page. You can create these controls using the `LiveNavigate` built-in function.

For information about setting up the `LiveNavigate` function, see “[LiveNavigate](#)” on page 405.

15.2.5 Allowing End Users to Navigate From the Table of Contents

If a Live document includes a table of contents, you can allow end users to navigate by holding down CTRL and clicking a heading in the table of contents. This functionality is enabled by the **Hyperlink** option in the properties of a table of contents placeholder, which also enables hyperlinks in the table of contents in PDF and HTML output. Enabling navigation from the table of contents can make it easier for end users to navigate longer Live documents.

To allow end users to navigate from the table of contents in a Live document:

1. In Design Manager, from the Library, drag the page that contains the table of contents placeholder to the Edit Panel.

The page opens in Designer.

2. Select the table of contents placeholder and click .

The **TOC Properties** dialog box opens.

3. Select the **Hyperlink** check box.

4. Click **OK**.

The **TOC Properties** dialog box closes.

5. From the Menu bar, select **File > Save**.

For more information about tables of contents in Live documents, see “[Using Tables of Contents and Indexes in Live Documents](#)” on page 66.

For information about creating a table of contents, see *Designing Customer Communications* in the Exstream Design and Production documentation.

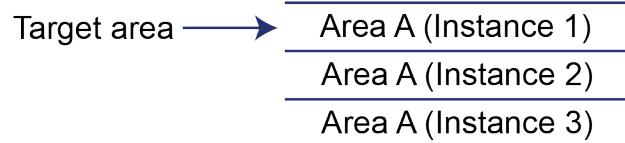
15.2.6 Adding Automatic Navigation to Specific Interactive Areas

To provide an intuitive, automated editing experience, you can add navigation features based on end-user selections and input that can guide end users to specific areas with which they need to interact. For example, if an end user selects a check box to indicate that a customer lives in the United States, you can have the cursor move immediately after the selection to a text box where the end user must enter additional information required for United States residents.

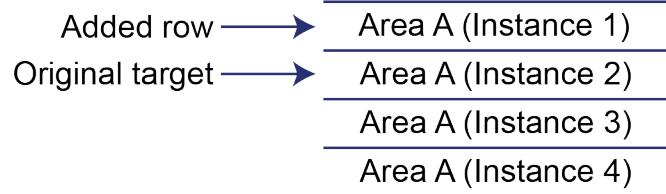
You can set up this functionality using the `LiveGoto` built-in function. Because the `LiveGoto` function might be triggered based on user selections or input and requires you to identify the navigation target, you must carefully plan and test a Live document that includes automatic navigation.

If you want to use the `LiveGoto` function to add automatic navigation to a `LiveDocument`, it is important to understand how the Live document's structure can affect the navigation instructions you set up. For example, suppose you want to set up the `LiveGoto` function so that the cursor automatically moves to the interactive area named "Area A" when an end user makes a specific selection. The table row containing "Area A" can be duplicated, and the table itself also can be duplicated. Therefore, depending on the end user's selections, "Area A" can potentially exist multiple times in the same customer. When you set up the `LiveGoto` function, you must specify the instance of "Area A" to which you want end users to navigate. As the following example illustrates, the instance number of the area to which you want to navigate can change as the document changes. In this example, assume the `LiveGoto` function specifies instance 1 of "Area A" as the target area. After a row is added, the original target area becomes instance 2, and when the `LiveGoto` function is executed, the cursor moves to Instance 1.

Navigation target before a row is added



Navigation target after a row is added



Therefore, if you want to include automatic navigation in a complex Live document, you must carefully plan and test your design so that you specify the correct instance of an area in the function.

For information about setting up the `LiveGoto` function, see ["LiveGoTo" on page 399](#).

15.2.7 Customizing the Outline Viewer

The Outline Viewer is a separate panel in LiveEditor that provides end users with an overview of the content and hierarchy of the Live document. To navigate through a Live document, an end user can click an object name in the Outline Viewer to go directly to that object. For example, if an end user clicks the Statement Summary Page in the Outline Viewer, that page appears in the design window. End users can also rearrange documents within a Live document by dragging and dropping them in the Outline Viewer, if you have enabled that feature in the Live setting object properties in Design Manager.

For more information about allowing users to rearrange documents in a Live Document, see [“Allowing End Users to Reorder Documents Using the Outline Viewer” on page 58](#).

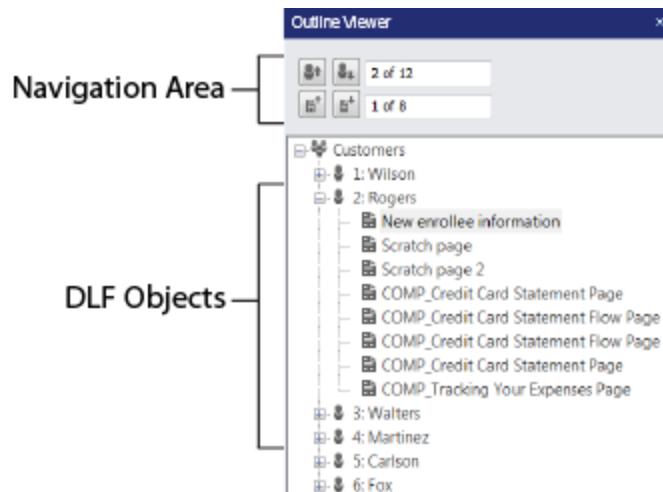
Keep in mind that the Outline Viewer in LiveEditor serves a different purpose from the Outline Viewer you can use in Designer as you design pages. Whereas the Outline Viewer in Designer provides a detailed list of all the objects on the page, the Outline Viewer in LiveEditor lists only high-level and Live objects.

If you include placeholder documents in the design of a Live document, end users can right-click an existing (non-placeholder) document in the Outline Viewer panel to access a context menu that allows the addition of a placeholder document that has not already been used for a particular customer in the Live document. Additionally, end users can right-click an existing placeholder document in the Outline Viewer panel to access a context menu that allows importing, updating, or appending external pages, as well as deleting the placeholder document.

For information about adding placeholder documents to a Live document, see [“Creating a Placeholder Document” on page 368](#).

You can also add filters to the Outline Viewer. End users can then select a filter to change the types of objects that appear in the Outline Viewer.

Example of the Outline Viewer in LiveEditor



This section discusses the following topics:

- “[Naming Objects in the Outline Viewer](#)” below
- “[Excluding Specific Objects from the Outline Viewer](#)” on the next page
- “[Creating Filters for the Outline Viewer](#)” on the next page

Naming Objects in the Outline Viewer

You can supply specific names for objects that will appear in the Outline Viewer. By giving objects meaningful names, you can help end users use the Outline Viewer to quickly locate pages in the Live document or areas that they need to edit. Because these names do not appear in the final output that is sent to customers, you can provide names that are meaningful to the end users, but not necessarily to the customer.

To supply a name that will appear in the Outline Viewer:

1. Open the object properties by doing one of the following:

To	Do this
Open the properties of a Design Manager object	From the Library, drag the object to the Property Panel.
Open the properties of a Designer object	Select the object and click  .

2. Click the **Interactive** tab.
3. In the **Live caption** box, enter the name that you want to appear in the Outline Viewer.
4. Do one of the following:
 - In Design Manager, from the Menu bar, select **File > Save**.
 - In Designer, from the Menu bar, select **Edit > Save**.

If you do not enter a name in the **Live caption** box, the Outline Viewer displays the name specified in the **Name** box on the **Advanced Properties** dialog box or, if that is not specified, the name specified in the **Reference name** box on the **Dynamic Size and Placement** tab. If you do not specify a name in any of these locations, the object type appears in the Outline Viewer. For example, for an editable text area, **Text** appears in the Outline Viewer.

If you are using the Live document to create an output that supports bookmarks, the Live caption information does not control the bookmark names, so the bookmarks will not necessarily match the Outline Viewer. However, in your fulfillment application, you can provide a variable that defines the bookmarks in the final output.

For information about using a variable to name bookmarks, see *Designing Customer Communications* in the Exstream Design and Production documentation.

Excluding Specific Objects from the Outline Viewer

If you do not want end users to browse to specific object, or if you want to streamline the Outline Viewer to make navigation easier, you can exclude specific objects from the Outline Viewer. For example, if you determine that end users do not need to navigate directly to a flow frame in your Live document, you can exclude the flow frame from the Outline Viewer. You could include only the necessary objects to make navigation as easy as possible.

Keep in mind that you cannot exclude specific pages or documents from the Outline Viewer.

To exclude a specific object from the Outline Viewer:

1. Open the object properties by doing one of the following:

To	Do this
Open the properties of a Design Manager object	From the Library, drag the object to the Property Panel.
Open the properties of a Designer object	Select the object and click  .

2. Click the **Interactive** tab.
3. Select the **Exclude from outline viewer** check box.
4. Click **OK**.
5. Do one of the following:
 - In Design Manager, from the Menu bar, select **File > Save**.
 - In Designer, from the Menu bar, select **Edit > Save**.

Creating Filters for the Outline Viewer

By default, only customers, documents, and pages appear in the Outline Viewer. However, you can create filters that allow end users to show hidden objects, sections, paragraphs, and interactive areas in the Outline Viewer. End users can select from the filters you provide to control the objects that appear. For example, a custom filter might be particularly helpful if your Live document contains many areas that can be hidden or visible. You can create a custom filter that an end user can use to see all the hidden areas in the document listed in the Outline Viewer. You might also create a filter that includes the interactive areas in a Live document so end users can use the Outline Viewer to navigate to each object that must be edited.

Keep in mind that by supplying a filter, you are not directly changing what appears in the Outline Viewer. Instead, you are allowing end users to change what appears in the Outline Viewer by applying the filter in LiveEditor.

To create a custom filter:

1. In Design Manager, from the Library, drag a view object associated with the Live document to the Property Panel.
 2. Click the **Outline Viewer Filters** tab.
 3. Under the **Available filters** box, click .
- The **Outline Viewer Filter** dialog box opens.
4. In the **Filter Name** box, enter a name for the filter (for example, Show All Interactive areas).
 5. To indicate the types of objects that appear in the Outline Viewer when an end user applies this filter, select from the following check boxes:

To	Do this
Show objects that are currently hidden in the Live document	Select Show hidden .
Show the sections in the Live document	Select Show sections .
Show the paragraphs in the Live document	Select Show paragraph .
Show the interactive areas in the Live document	Select Show editable areas .

6. Click **OK**.

The **Outline Viewer Filter** dialog box closes.

7. If you want LiveEditor to apply a default filter when the end user opens the Live document, click  in the **Default filter** box and select a default filter.
8. From the Menu bar, select **File > Save**.

Keep in mind that in order to make these filters available in a Live document, you must include the view object in the setting object associated with the Live document.

For more information about adding a view to a setting object, see ["Associating Views With a Live Document" on page 232](#).

Chapter 16: Controlling the Formatting Options Available to End Users

One way to allow end users to customize Live documents is to allow them to format text. You can set up a Live document so that end users can format both text they can edit and text they cannot change. If you choose to allow end users to format text, you can restrict the formatting options available to them if needed. For example, you can restrict the types of fonts that can be used in a Live document so that all text in the document uses fonts approved by the corporate branding strategy.

For more information about allowing end users to change and format text, see “[Creating a Basic Live Document](#)” on page 38.

You can use the following methods to control the formatting options available to end users:

Formatting control method

Type of formatting control	Description of method
Controlling font behavior	You can exert a wide range of control over the types of fonts and the formats of fonts available to end users in LiveEditor. For example, you can specify each point size of a font that an end user can use, as well as other formatting options such as bold or italic. On the other hand, you can choose not to restrict fonts, and allow end users to use fonts from their computer in any format.
Restricting colors	You can use color families to restrict the colors available to end users in LiveEditor. The colors in the color families you include in the Live document can then be applied to any objects for which end users have editing permissions.
Restricting specific formatting controls, such as the ability to make text bold	You can use views to disable specific formatting controls in LiveEditor. Views are objects that allow you to customize the appearance of the LiveEditor interface, as well as the availability of the features found on the interface. Views allow you granular control over each menu option in LiveEditor, making it possible for you to prevent end users from taking specific actions in a particular Live document. For example, you can use views to remove specific formatting options, such as the ability to change a font or make text bold. For information about using views, see “ Using Views to Customize the End-User Interface ” on page 226.
Providing pre-set formatting options	You can use style sheets to provide pre-set formatting options to end users. Style sheets allow end users to quickly and easily apply pre-approved fonts and formatting to any text that they can edit.

This chapter discusses the following topics:

- “[Controlling Font Behavior in Live Applications](#)” on the next page
- “[Restricting the Colors Available to End Users](#)” on page 257
- “[Using Style Sheets to Provide Pre-Set Formatting Options](#)” on page 258

16.1 Controlling Font Behavior in Live Applications

When you design Live documents that allow end users to change text content, you can also allow them to change text formatting, including the fonts that are used. As you specify font behavior in Design Manager, keep in mind that you must consider not only how the fonts will appear in LiveEditor, but also how they will be processed by the engine during packaging. For example, you can prevent any fonts that are not part of your corporate font library from being available for use. Conversely, if your company does not use a corporate font library, you can ensure that fonts are available in LiveEditor by embedding them as part of the DLF file during packaging.

Before specifying font settings for your Live document, you should familiarize yourself with the three methods you can use to control font behavior in Exstream Live:

- **Referencing fonts:** You reference fonts through settings on the Live setting object. When you reference fonts, you are including in the DLF file all information about the fonts used in your Live application. If end users do not have referenced fonts installed on their computers, then the missing font(s) will be replaced by the default font on their system.
- **Including fonts:** You include fonts through settings on the DLF output object. When you include fonts, you are adding basic information to the DLF file about fonts that are not used in your Live application but that may be needed during fulfillment. The information includes such things as character size and kerning details for the extra fonts.
- **Embedding fonts:** You embed fonts through settings on the Live setting object. When you embed fonts, you are packaging the fonts themselves as part of the DLF file. When end users open a DLF that contains embedded fonts, those fonts are temporarily installed on their computers and are available when they edit the Live document. Keep in mind, however, that certain fonts are subject to licensing restrictions and cannot be embedded.

For more information about using licensed fonts in a Live document, see “[Embedding or Referencing Fonts in a Live Document](#)” on page 251.

Note: If you have enabled complex text layout for your database (by selecting the **Enable complex text layout** check box on the **Text and Fonts** tab in the **System Settings** in Design Manager), then you must think carefully about the complex text fonts that you use in your Live documents. Embedded fonts will cause a significant increase in the size of the DLF file; therefore, you should not embed complex text fonts as part of a Live document. This means, however, that the fonts in the Live document will not appear as expected if they are not available to end users on their local machines.

For information about using complex text languages in your Live application, see “[Creating Live Documents for Complex Text Languages](#)” on page 69.

Because font needs might be different in your interactive environment than they are in your design environment, the font settings on the Live setting object supersede font settings elsewhere in your database. For example, even though a font might be restricted on the font object in Design Manager (under the **Environment > Design > Fonts** heading), those settings will be ignored if they contradict font settings on the Live setting object.

This section discusses the following topics:

- “[Controlling the Fonts that End Users Can Use](#)” below
- “[Controlling How Fonts are Handled When Packaging a DLF File](#)” on page 251

Tip: In addition to controlling font usage using these methods, you can also use style sheets to control fonts and formatting. You can determine which method, or mixture of methods, is best for your organization.

For information about using style sheets, see “[Using Style Sheets to Provide Pre-Set Formatting Options](#)” on page 258.

16.1.1 Controlling the Fonts that End Users Can Use

As you work with the font settings on the **Design** tab of the Live setting object, you will need to distinguish between two types of font restrictions: fonts and font faces. In this context, "font" refers to the specific combination of font family, font size, and formatting. For example, Times New Roman Italic size 8 is one font and Times New Roman Bold size 8 is another font. "Font faces" refer to a font family, including all sizes and formatting. Times New Roman is one font face and Arial is another font face.

To define which fonts end users can use when they edit the text in a Live document:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
The setting object opens for you to define.
2. Click the **Design** tab.
3. To control the types of fonts end users can use in the Live document, complete one of the following sets of steps:

To	Do this
Allow end users to use fonts located on their computers	<p>a. From the Allowed fonts drop-down list, select All fonts (in the DLF).</p> <p>b. From the User added fonts drop-down list below the box, select one of the following options:</p> <ul style="list-style-type: none"> • Cannot add fonts—End users cannot add their own fonts. • Can add fonts, honor restrictions—End users can add their own fonts but must honor the restrictions on fonts that you specify on the font object properties. For example, if Arial size 10 and 12 are set up in the database, then end users cannot use Arial size 14 in LiveEditor. Select this option if you want the fonts that end users add to adhere to the restrictions set up on fonts in your design database. • Can add fonts, ignore restrictions—End users can add their own fonts if they are also set up in your design database. For example, if Arial size 10 and 12 are set up in the database, then end users can use any size or formatting of Arial fonts in LiveEditor. Select this option if you want to ensure that end users will use the same types of fonts as documents produced from Exstream Design and Production. • Can add font faces—End users can add any font faces. <p>For information about setting restrictions on fonts, see <i>System Administration</i> in the Exstream Design and Production documentation.</p>
Allow end users to use only specified fonts (in the point size and format you specify)	<p>a. Select Listed fonts.</p> <p>b. Below the box, click  .</p> <p>The Select Font dialog box opens.</p> <p>c. Do one of the following:</p> <ul style="list-style-type: none"> • Select the Show only fonts in the database check box to limit the list to only those fonts or font faces that are loaded in the database. • Clear the Show only fonts in the database check box to show all fonts available on your system. <p>d. Select the font, point size or sizes, and any formatting options you want to allow end users to use.</p> <p>e. Click OK.</p> <p>The Select Font dialog box closes and the font (including variations of point sizes and formats) appears in the box.</p> <p>f. Repeat step b through step d to specify as many fonts as needed.</p>

To	Do this
Allow end users to use only specified fonts (in any size and format)	<p>a. Select Listed font faces.</p> <p>b. Below the box, click  .</p> <p>The Select Font dialog box opens.</p> <p>c. Do one of the following:</p> <ul style="list-style-type: none"> • Select the Show only fonts in the database check box to limit the list to only those fonts or font faces that are loaded in the database. • Clear the Show only fonts in the database check box to show all fonts available on your system. <p>d. Select the font that you want to allow end users to use.</p> <p>e. Click OK.</p> <p>The Select Font dialog box closes and the font appears in the box.</p> <p>f. Repeat step b through step d to specify as many fonts as needed.</p>

4. From the Menu bar, select **File > Save**.

16.1.2 Controlling How Fonts are Handled When Packaging a DLF File

Because the fonts installed on end users' systems might be different from the fonts installed on your system, you can adjust various settings to specify which fonts and font properties will be included when you package a DLF file.

This section discusses the following topics:

- “[Embedding or Referencing Fonts in a Live Document](#)” below
- “[Managing Font Resources in the DLF Output Object](#)” on page 254

Embedding or Referencing Fonts in a Live Document

As part of the packaging process, you can make fonts available in LiveEditor by referencing or embedding them in a Live document. A referenced font requires that the same font be available on an end user's system, whereas an embedded font is packaged as part of the DLF file itself. You can also control how embedded fonts with licensing restrictions are handled in LiveEditor.

Because fonts are generally distributed in the same manner as software—that is, they are licensed rather than sold outright—many font licenses place restrictions on how fonts can be used. For example, while some fonts cannot be embedded as part of a file, others can be embedded but not edited—instead, they are used only for the purposes of printing or previewing a file. Conversely, fonts without licensing restrictions can be edited, printed, and added to end

users' systems without limitations. For information about the licensing restrictions of a particular font, contact the vendor that supplied the font.

As you work with fonts for Live documents, keep in mind that even though LiveEditor replaces missing fonts during an editing session, it retains the font metrics (such as spacing, width, and height) for each missing font. For example, suppose you design a Live document that references Futura font, but an end user does not have Futura on his or her system. For that editing session, LiveEditor will temporarily replace Futura with the default system font (usually Arial), but it will retain the Futura font metrics. Futura will still show up in the list of fonts in LiveEditor, but it will not be available for editing the Live document.

The following example shows how text appears when a missing font is replaced by Arial in a Live document:

*Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam hendrerit
 tortor at justo condimentum ac iaculis arcu iaculis.*

LiveEditor does not permanently replace a missing font, because other end users who have the missing font on their systems might also want to make changes to the same DLF file.

Depending on your design, end users might replace a missing font with another font altogether. To avoid confusion, you should use only fonts in your design that are readily accessible to end users.

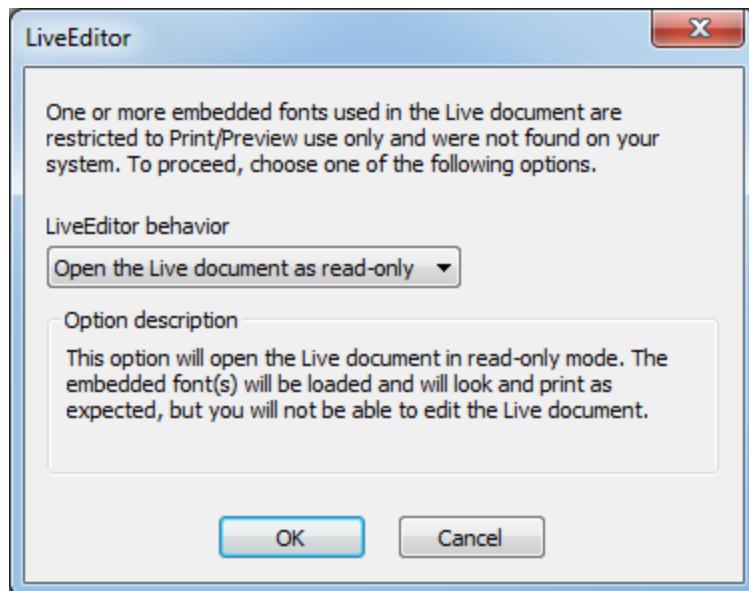
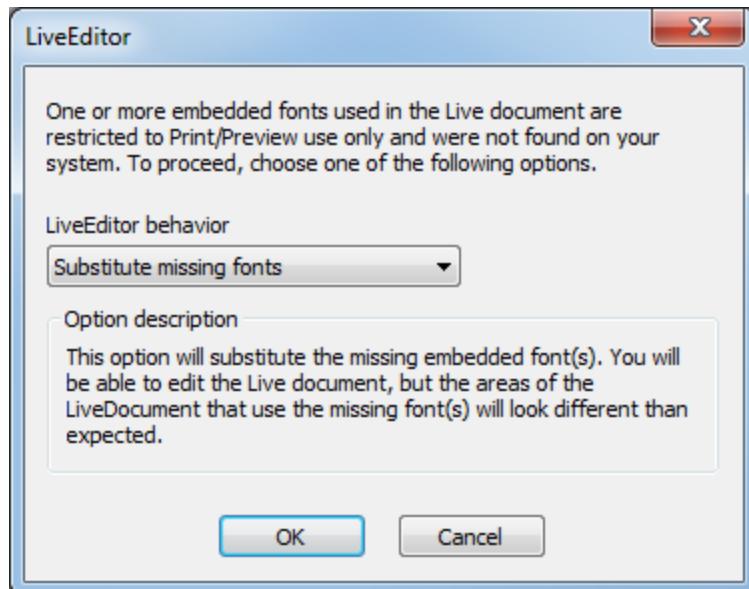
Note: If you have enabled complex text layout for your database (by selecting the **Enable complex text layout** check box on the **Text and Fonts** tab in the **System Settings** in Design Manager), then you must think carefully about the complex text fonts that you use in your Live documents. Embedded fonts will cause a significant increase in the size of the DLF file; therefore, you should not embed complex text fonts as part of a Live document. This means, however, that the fonts in the Live document will not appear as expected if they are not available to end users on their local machines.

For information about using complex text languages in your Live application, see “[Creating Live Documents for Complex Text Languages](#)” on page 69.

Changing the Font Packaging Properties for a Live Document

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
The setting object opens for you to define.
2. Click the **Resources** tab.
3. In the **Font packaging options** section, select an option from the **Packaging behavior** drop-down list:

- **Reference all fonts**—If you select this option, no design fonts will be included when you package the DLF. The fonts will be referenced from the end users' systems. If end users do not have the design fonts on their systems, then the missing fonts will be replaced by the default font on their system.
 - **Embed all fonts**—If you select this option, all design fonts will be included when you package the DLF. This ensures that the fonts will be available to all end users, even if they do not have the fonts installed on their systems. Keep in mind, however, that embedded fonts might be subject to font licensing restrictions. If you want to include any additional fonts or font faces that you added in the **Allowed fonts** section on the **Design** tab, select the **Embed listed fonts** check box.
4. If you selected **Embed all fonts** from the **Packaging behavior** drop-down list, then the **Restricted font behavior** drop-down list becomes available. Select one of the following options to specify how restricted fonts will be handled when end users open a Live document:
- **Make the DLF read-only**—This setting opens the Live document in read-only mode. End users will not be able to make changes to the Live document.
 - **Substitute restricted fonts**—This setting substitutes the missing fonts with the default font on the end user's system.
 - **Prompt the user**—This setting prompts end users to select one of the other two options (opening in read-only mode, or substituting restricted fonts). End users will see the following options in LiveEditor:



5. From the Menu bar, select **File > Save**.

Managing Font Resources in the DLF Output Object

Basic information about certain fonts associated with the Live application is included in the DLF file automatically. The basic information includes such things as character size and kerning details. By default, if end users have those fonts installed on their systems, they can use the

fonts to make sure their changes to the Live document have a consistent appearance with the rest of the document.

Basic information about the following fonts is automatically packaged and sent to end users:

- The font specified in the **Default font** box on the **Text and Fonts** tab of the **System Settings**
- All fonts used in the bullets set on the **Text and Fonts** tab of the **System Settings**
- The font specified in the **Default footnote font** box on the **Documents** tab of the application properties
- The font specified in the **Default font for text and tables** box and the fonts used in the bullets that are specified on the **Design Defaults** tab of the page template properties (if the Live document uses a page template)
- The fonts included in the **Font** box and the fonts used in the bullets that are specified on style sheet properties in Design Manager

You can change the font settings on the DLF output object to allow fonts to appear in a Live document that are not used in a Live application but might be needed during fulfillment. For example, suppose you have a font in your database that is not used in a particular Live document. However, you want to give end users the ability to import into the Live document formatted RTF text that uses that font. After end users import the RTF text and submit the Live document for fulfillment, the font settings on the DLF output object determine how that font is handled during fulfillment. These settings do not embed fonts in the DLF; instead, they include information about the specified fonts—such as character size and kerning details—for all outputs in an application.

For all fonts other than the ones automatically packaged and sent to end users, you can control whether they are included in the Live document. To control the fonts included in the DLF file:

1. In Design Manager, from the Library, drag the output object associated with the Live document to the Property Panel.

The output object opens for you to define.
2. Click the **Resource Management** tab.

3. From the **Include additional fonts** drop-down list, select one of the following options:

To	Do this
Include font resources only for the fonts used in the application	Select None .
Include all of the fonts used in the application, in addition to any fonts in the database that are designated as Live custom fonts. (Live custom fonts allow end users to use fonts during editing that are not already present in the DLF file.)	Select Live custom fonts .
Include all of the fonts available in the database	Select All library fonts .

4. From the Menu bar, select **File > Save**.

Creating Live Custom Fonts

Live custom fonts allow end users to use fonts during editing that are not already present in the Live document. These fonts are part of your database, but are not used in the Live application. For example, if the document uses only variations of the Times New Roman font, but you want to let users use an Arial font, you can define Arial as a Live custom font. You use the DLF output object to control whether Live custom fonts are included in the DLF file.

1. In Design Manager, in the Library, go to **Environment > Design > Fonts**.
2. Right-click the **Fonts** heading and click **Import New Font**.
3. In the **Select Font** dialog box, select a font from the **Font** list. The list contains all of the fonts that are available to your workstation.
4. Click **OK**.
5. In the Property Panel, configure the font's attributes to meet your company's specifications using the following options:

To	Do this
Add additional information about a font, such as how it is used (optional)	In the Description box, enter a description.
Use magnetically scannable ink, which is often used on checks	In the Special options list, select Use MICR ink .

To	Do this
Import variations of a font (including specific point sizes and/or effects)	<p>a. Below the Font list box, click .</p> <p>b. In the Select Font dialog box, select font, font size, and font format options as needed. To specify a range of font sizes, select the Add multiple sizes check box.</p> <p>You must include all variations of a font in order for it to be available in LiveEditor as a custom font. For example, if you want end users to be able to use any size of Arial font in italics, you must include in the Font list box each available Arial size with italic formatting specified on each size.</p> <p>c. Click OK.</p> <p>The Select Font dialog box closes and the font variation or variations that you selected appear in the Font list box.</p>
Import the font metrics for the sizes and styles of fonts from your workstation font directory. You might want to refresh font metrics because the code page or operating system version has changed.	In the Font list box, select a font and click Refresh Metrics .

16.2 Restricting the Colors Available to End Users

You can use the Live setting object to define which colors end users can use when they edit text or objects in a Live document. Restricting the colors available to end users allows you to give end users the freedom to format and change the Live document, but you can still enforce adherence to corporate color rules.

To restrict the colors available to end users:

1. In Design Manager, from the Library, drag the setting object associated with the Live document to the Property Panel.
The setting object opens for you to define.
2. Click the **Design** tab.
3. From the **Allowed color families** drop-down list, select one of the following options:

To	Do this
Allow end users to select any color	Select None .

To	Do this
Allow end users to use any colors that are present in color families in the design database	Select All .
Allow end users to use colors present in only specified color families	<p>a. Select Listed.</p> <p>b. Below the box, click  .</p> <p>The Select Color Family dialog box opens.</p> <p>c. Select the color family from which you want to allow end users to use colors and click OK.</p> <p>The Select Color Family dialog box closes and the color family you selected appears in the box.</p> <p>d. Repeat step b and step c to specify as many color families as needed.</p>

For more information about color families, see *System Administration* in the Exstream Design and Production documentation.

- From the Menu bar, select **File > Save**.

16.3 Using Style Sheets to Provide Pre-Set Formatting Options

In Exstream Live, you can use style sheets to enforce corporate standards for the end users of a Live document. When you use style sheets in a Live document, end users can select which style sheet they want to use (if you have provided multiple style sheets), and then use a drop-down list to apply styles from that style sheet to selected text.

For more information about style sheets, see *System Administration* in the Exstream Design and Production documentation.

You can also use the `LiveSetStyleSheet` built-in function to automatically associate a style sheet with a Live document.

By default, when an end user changes style sheets, style changes occur at the page level. In other words, the end user must select a style sheet for each page in a Live document. If you want style sheets to apply style changes at the customer level and to all objects, you must select the **Styles are applied to all pages and objects** check box on the setting object associated with the Live document. When this check box is selected, styles are applied at the customer level whether you are using the menu option in the DLF or using the `LiveSetStyleSheet` function to set a style sheet.

For information about setting up the `LiveSetStyleSheet` function, see “[“LiveSetStyleSheet” on page 418](#)”.

To use style sheets to control formatting:

1. In Design Manager, from the Library, drag the setting object associated with the Live document to the Property Panel.

The setting object opens for you to define.

2. Click the **Design** tab.
3. From the **Allowed style sheets** drop-down list, select one of the following options:

To	Do this
Provide end users with no style sheets	Select None .
Allow end users to use any style sheet defined in the design database	Select All .
Allow end users to use only specified style sheets	<ol style="list-style-type: none"> Select Listed. Below the box, click  . The Select Style Sheet dialog box opens. Select the style sheet that you want to provide to end users and click OK. The Select Style Sheet dialog box closes and the style sheet that you selected appears in the box. Repeat step b and step c to specify as many style sheets as needed. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Note: If you create a Live document that uses a style sheet that is not included in the Allowed style sheets list, then the styles specified by the style sheet will be visible when the end user opens the Live document. However, the style sheet itself will not be available for end users to reapply to the Live document. If end users apply a style sheet from the Allowed Style Sheets list, then they will not be able to apply the style sheet that was used to create the Live document.</p> </div>

4. If end users will return this Live document for fulfillment, you must complete the following steps:

- a. Make sure that the style sheets specified in the fulfillment application matches those included in the design application.
- b. Use the **USE_DLF_STYLE** engine switch to disable the use of the **SYS_StyleSheet** variable as a formula in a fulfillment application.

For more information about the **USE_DLF_STYLE** engine switch, see “[“USE_DLF_STYLE” on page 442](#)”.

For more information about defining resources for Live fulfillment, see “[“Defining Application Resources for Live Document Processing” on page 379](#)”.

5. If you want style sheets to apply at the customer level and to all objects, select the **Styles are applied to all pages and objects** check box.
6. From the Menu bar, select **File > Save**.

Chapter 17: Controlling How End Users Save and Print Live Documents

End users have the ability to save a Live document in its native format to ensure that changes they have made are not lost. However, end users can also save a Live document as a PDF file. The ability to save a Live document as a PDF supports workflows in which end users email a copy of the Live document they filled out while on the phone with a customer, or workflows in which end users archive a copy of the Live document after it has been completed. Exstream allows you to enable or restrict the end users' ability to save specified Live documents as PDFs. If you choose to allow end users to save Live documents as PDFs, you also have control over some of the format and appearance options in the produced PDF and over the security options used in the PDF.

You can also control whether end users can print Live documents from LiveEditor after they have made changes to them. For example, if end users will use the Live document to enter customer data that is used only to drive the creation of an application, you might choose to restrict the end users' ability to print the Live document locally to protect the customer data. On the other hand, if your workflow is designed so that end users will print a copy of the edited Live document from their computer and send it to the customer, you can allow them to print locally.

Tip: As you design Live documents that end users will print locally, keep in mind that the print quality for imported PDFs is lower when you print directly from LiveEditor. In order to accommodate display on a monitor, LiveEditor converts imported PDFs into rasterized images and then displays those images on the screen. Likewise, LiveEditor uses rasterized images of imported PDFs when creating printed versions of Live documents. When creating PDF versions of Live documents that contain imported PDFs, however, LiveEditor uses the original imported PDF files. As a result, if end users want to print a Live document that contains imported PDF pages, they should first use the **Save As PDF** option from the LiveEditor Menu bar (**Tools > Save As PDF**) and then print the resulting PDF.

Another way to provide guidance to end users is to remove the **Print** option from the LiveEditor Menu bar. With printing disabled from within LiveEditor, you can add a "Save this document as PDF" button to the Live document and tie the button to the `LiveSaveAsPDF` function. End users can then print the resulting PDF.

For more information about the `LiveSaveAsPDF` function, see "["LiveSaveAsPDF" on page 410](#)".

For more information about removing options from the LiveEditor Menu bar, see "["Using Views to Customize the End-User Interface" on page 226](#)".

This chapter discusses the following topics:

- "["Controlling End User Options for Saving a Live Document as a PDF" on the next page](#)"
- "["Defining the Properties of PDFs Created from Live Documents" on the next page](#)"
- "["Controlling How End Users Print a Live Document Locally" on page 264](#)"

The information about allowing end users to print and save recipient copies of a customer's document(s) as PDFs is discussed in a separate task.

For information about adding recipients to a Live document and controlling how those copies are saved or printed, see "[Allowing End Users to Add and Modify Copies of Documents](#)" on [page 178](#).

17.1 Controlling End User Options for Saving a Live Document as a PDF

By default, the option to save a Live document as a PDF in LiveEditor is enabled and end users can access it through the LiveEditor menus and toolbars. However, you can disable these options or provide end users with an automatic way to save Live documents.

To control end users options for saving a Live document as a PDF, complete the appropriate task in the following table:

To	Do this
Disable saving options so end users cannot save Live documents as PDFs	<p>Use the Live document's view object to remove the "Save as PDF" option from the menus and toolbars.</p> <p>For more information about customizing the view object, see "Setting Up a View to Remove or Add Options From the End-User Interface" on page 227.</p>
Provide end users with an automatic way to save a Live document, such as a <code>Save As PDF</code> button on a page	<p>Use the <code>LiveSaveAsPDF</code> built-in function to add automated printing capabilities to the Live document.</p> <p>For more information about setting up the <code>LiveSaveAsPDF</code> built-in function, see "LiveSaveAsPDF" on page 410.</p>

If you choose to allow end users to save a Live document as a PDF, you can also control many properties of the generated file.

For more information about controlling the properties of PDFs created from Live documents, see "[Defining the Properties of PDFs Created from Live Documents](#)" below.

17.2 Defining the Properties of PDFs Created from Live Documents

Allowing end users to create PDFs from Live documents gives them flexibility in the methods they can use to share the content of the Live document. For example, they can easily email it to a customer or other employee who does not have LiveEditor installed. In addition, saving a Live document as a PDF allows end users to leverage many of the security features available in PDF format.

To define the properties of PDFs created from Live documents:

1. In Design Manager, from the Library, drag the setting object associated with the Live document to the Property Panel.
2. Click the **Resources** tab.
3. Use the options in the **Save as PDF options** area to control the format and appearance of the PDF that will be created:

To	Do this
Specify the sharpness of text and graphics in the generated PDF. (Sharpness of text and graphics is measured in dots per inch (dpi).)	<p>From the Resolution drop-down list, select the dpi to use in the generated PDF.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Tip: Make sure to use the same resolution setting as the objects on the design page use; these resolution settings can contradict each other and cause quality issues if they are not compatible. </div>
Specify whether the generated PDF is set up for duplex printing	<p>From the Simplex / duplex drop-down list, select one of the following options:</p> <ul style="list-style-type: none"> • Simplex—Sheets of paper are always printed on one side. • Duplex—Sheets of paper are always printed on both sides.
Specify the color settings in the generated PDF	<ol style="list-style-type: none"> a. From the Color mode drop-down list, select one of the following options: <ul style="list-style-type: none"> • Black and white—The PDF is produced in black and white only. • Gray scale—The PDF is produced in shades of gray. • Highlight—The PDF is produced in black, white, and highlight color (s) only. • Full color—The PDF is produced in full color. On printers that do not print in full color, patterns simulate different colors. b. If you selected Highlight, use the Highlight color color well to define the secondary color used for images.
Compress the generated PDF to make it smaller. When a Live document is saved as a PDF, the default compression level of 6 is always used.	Select the Compress check box.

4. If you want to add security features to the PDF to prevent other users from making changes to it or interacting with it in certain ways, use the options in the **PDF security - Do not allow** area:

To	Do this
Prevent any user of the PDF from printing any part of the document	Select the Printing check box.

To	Do this
Prevent any user of the PDF from highlighting and copying content	Select the Changing the document check box.
Prevent any user of the PDF from editing text or graphics in the file	Select the Selecting text and graphics check box.
Prevent any user of the PDF from editing form fields or adding new. However, users can fill in existing fields on the form.	Select the Adding or changing annotations or form fields check box.

- From the Menu bar, select **File > Save**.

17.3 Controlling How End Users Print a Live Document Locally

The printing options in LiveEditor provide end users with the flexibility to print a range of pages, a selection of pages, a specific document in the Live document, a specific customer in the Live document, or the entire Live document. By default, these printing options are enabled and end users can access them through the LiveEditor menus and toolbars. However, you can disable these printing options or provide end users with an automatic way to print Live documents.

To control how end users print a Live document locally, complete the appropriate task in the following table:

To	Do this
Disable printing options so end users cannot print Live documents locally	Use the Live document's view object to remove the printing options from the menus and toolbars. For more information about customizing the view object, see "Setting Up a View to Remove or Add Options From the End-User Interface" on page 227 .
Provide end users with an automatic way to print, such as a Print button on a page	Use the LivePrint built-in function to add automated printing capabilities to the Live document. For more information about setting up the LivePrint built-in function, see "LivePrint" on page 407 .

Chapter 18: Setting Up Tools to Review and Validate End Users' Changes

You can make review tools available in Exstream Live to help end users review their work before it is sent to the next stage in the workflow. You can also validate the data entered by end users.

The following review tools are available for use in LiveEditor:

- **Spell check, grammar check, excluded word check**—These tools allow end users to run spell check, grammar check, and excluded words check on text in a Live document. You can also use the LiveCheck built-in function to automate and/or require these checks.
- **Revision tracking and commenting**—The text revision tools help to track changes to a Live document so that approvers know exactly what changes have been made and whether suggestions have been incorporated. You can also require end users to add comments to explain each revision.
- **Variable validation**—Using validation variables lets you check the data that end users add to a Live document. You can use variables to require a specific format for the data entered, and you can customize the warnings that end users receive when they enter incorrect data.

This chapter discusses the following topics:

- “[Setting Up Text Proofing Tools](#)” below
- “[Setting Up Live Document Review Tools](#)” on page 267
- “[Validating Data Entered by End Users](#)” on page 284

18.1 Setting Up Text Proofing Tools

You can check the content quality of a document by running spell check, grammar check, and an excluded words check. Before you begin using the spelling and excluded words checks, however, you must first set up dictionaries in Design Manager. You can create custom spelling and excluded words checks to prevent unnecessary error flags, but the grammar check is standard with Exstream.

For more information about setting up dictionaries in Design Manager, see *Designing Customer Communications* in the Exstream Design and Production documentation, and *System Administration* in the Exstream Design and Production documentation.

This section discusses the following topics:

- “[Specifying Behavior for Spelling, Grammar, and Excluded Word Check](#)” below
- “[Supplying Custom Dictionaries for End Users](#)” on the next page

18.1.1 Specifying Behavior for Spelling, Grammar, and Excluded Word Check

You can control whether spell and grammar checking is active when end users open a Live document. For example, if you create a Live document that contains a large number of pages, you might want to turn off spell checking and/or grammar checking to reduce the amount of time it takes the Live document to load in LiveEditor. However, unless you remove spell checking and grammar checking from the menu commands available in LiveEditor, end users can still access and run spell checker after they open the Live document.

For more information about adding or removing commands from the LiveEditor Menu bar, see [“Customizing the End User Interface” on page 226](#).

Specifying the Initial Behavior for Spelling and Grammar Check

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Spell checking active on open** and the **Grammar checking active on open** drop-down lists, select one of the following options:
 - **On**—Spell checking and or grammar checking will be turned on when end users open a Live document.
 - **Off**—Spell checking and or grammar checking will be turned off when end users open a Live document.
 - **Last use**—Spell checking and or grammar checking will be turned on or off depending on which spell or grammar checking settings the end user selected during his or her previous LiveEditor session.
4. From the Menu bar, select **File > Save**.

Automating Spelling, Grammar, and Excluded Word Check

You can use the LiveCheck function to automatically check all or part of your Live document for spelling and grammatical errors, and for excluded words (if your company has an excluded

words list). You can also use the LiveCheck function to require end users to run any or all of those checks.

For more information about using the LiveCheck function, see “[Live Built-In Functions](#)” on [page 381](#).

18.1.2 Supplying Custom Dictionaries for End Users

Spelling dictionaries are referenced when end users run spell check. You can define which types of spelling dictionaries are allowed or must be used by end users. Controlling the use of spelling dictionaries can help you enforce corporate standards or ensure that end users do not introduce common errors when they edit the Live document.

You can define the use of spelling dictionaries in the following ways:

To	Do this
Allow end users to use local dictionaries when they run spell check (These are dictionaries in which individual end users have made modifications by right-clicking a misspelled word.) When you select this option, end users can use any of the following dictionaries: <ul style="list-style-type: none">• A local dictionary• A local dictionary and the custom Exstream spelling dictionary• Only the custom Exstream dictionary	<ol style="list-style-type: none">1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.2. Click the Editor Framework tab.3. In the Spelling and grammar options section, select the Allow user spelling dictionaries check box.
Require end users to use the custom Exstream spelling dictionaries when they run spell check	<ol style="list-style-type: none">1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.2. Click the Editor Framework tab.3. In the Spelling and grammar options section, select the Require Exstream spelling dictionaries check box.

18.2 Setting Up Live Document Review Tools

Revision tracking lets multiple users collaborate on changes that are made to a Live document. Using revision tracking, end users can easily make and view tracked changes in LiveEditor as they work. Additionally, end users can add comments to a revision to explain why a change was made. Designers can use the XML history file to see a complete history of the changes that were made to a Live document. Revision tracking tracks additions or deletions to the text, but keep in mind that formatting changes are not tracked. Furthermore, end users can comment on, but not alter, charts and graphics.

In addition to revision comments, you can use revision notes to require end users to describe the changes that they have made to a Live document as a whole. Revision notes can be used in conjunction with, or as an alternative to, revision comments, which apply to individual revisions.

Revision tracking lets you monitor changes to a Live document in both simple and complex workflows. For example, you can send a Live document to a single end user, and then use revision tracking to track any changes he or she makes. But revision tracking can be particularly valuable when used as part of a larger workflow, allowing many end users to collaborate and comment on changes that were made to a Live document.

For instance, suppose that you are a contract manager for a financial services company, and you are creating a new loan contract proposal for a client. You are using a DLF template as the basis for the custom contract, but you require input from others when setting up the contract. To track changes to the document from beginning to end, you set up the application in Design Manager so that XML history and revision tracking are turned on for all users. Turning on XML history and revision tracking ensures that every change made to the Live document will be recorded with a date, time, and user stamp. You also configure the Live objects in Designer so that end users must enter a comment to explain each change they make. After you set up the template, you can create a DLF file. Then, after logging in to LiveEditor with your user credentials, you can open the Live document and review the contract. You make several changes, including the interest rate and the repayment terms. These changes are automatically logged in the XML revision history file. The items included in the history file include the object names, the text changes, your user name, the date, and the time. You can then send the Live document (as a PDF) to the customer who responds with several changes. You can open the Live document and make the changes the customer has suggested. Again, the XML history file captures the changes to the Live document. With the changes that the customer has suggested, the DLF file is ready for approval by your manager. You can then send the Live document to your manager, who opens it in LiveEditor, reviews the revision history, and accepts the changes that you made. The acceptance of the changes is recorded in the XML history file, along with your manager's name, the date, and the time. After you receive approval, you can submit the loan contract proposal for fulfillment.

You can also produce output that contains the original version, current version, or marked up version of the Live document that contains tracked changes. For example, you might want to produce a marked up version of a loan contract proposal that shows proposed changes so that a client can see them before they are approved, or you might want to produce the original version of the document for archival purposes. To specify the version of the document that you want to produce during fulfillment, you must use the SHOWREVISION engine switch. Keep in mind that, by default, the engine produces the current version (including unapproved changes, without markup) during fulfillment.

For more information about the SHOWREVISION engine switch, see “[SHOWREVISION](#)” on page [441](#).

The following table details how revision tracking and commenting behave in certain interactive areas:

Interactive area	Behavior of revisions and comments
Text/text box	Revisions and comments are allowed on text and text boxes.
Table	Revisions and comments are allowed on tables and table rows.
Variable	<ul style="list-style-type: none">Revisions and comments are allowed on plain text and RTF text placeholder variables. All other placeholder variables do not allow revisions or commenting. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"><p>Note: If end users replace the value of a plain text or RTF text placeholder variable, then any previous revisions will not show up in the Current revisions list on the Revision Viewer palette in LiveEditor.</p></div> <ul style="list-style-type: none">Revisions and comments are allowed on formatted text variables.Images populated by placeholder variables allow comments only.Revisions and comments are not allowed on formula variables.Revisions and comments are not allowed on tagged text variables.
Image	Comments are allowed on images, but revisions are not.
Chart	Comments are allowed on charts, but revisions are not.

This section discusses the following topics:

- [“Using Revision Tracking to Monitor Changes Made to a Live Document” below](#)
- [“Setting Up Revision Commenting to Capture Explanations for Changes” on page 271](#)
- [“Tracking Revisions and Revision Comments in LiveEditor: The End User Experience” on page 274](#)

18.2.1 Using Revision Tracking to Monitor Changes Made to a Live Document

Revision tracking lets you follow changes to a Live document throughout its life cycle. You can track changes made to a Live document by a single user, but revision tracking is an especially useful tool when multiple end users are opening and editing the same DLF file. You can also use XML history tracking to keep a record of all changes made to a Live document. Depending on the settings you specify in Design Manager, you can keep a record of as many or as few revisions as you choose.

This section discusses the following topics:

- “[Controlling How Revisions Are Tracked](#)” below
- “[Setting Up XML History Tracking for Variables](#)” below
- “[Specifying the Number of XML Revisions to Keep](#)” on the next page

Controlling How Revisions Are Tracked

The **Revision tracking** options on the Live setting object let you specify whether changes to a Live document are tracked. If you have a workflow that requires approval for all changes, tracking changes can make it easier for approvers to know which edits were made and who made the edits. To see current revisions and revision history in LiveEditor, end users use the **Revision Viewer** palette, accessed by selecting **View > Revision Viewer** from the Menu bar in LiveEditor.

To control how revisions are tracked:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Revision tracking** drop-down list, select one of the following options:
 - **User selectable**—End users decide whether they want to track the changes they make. The Review toolbar is available in LiveEditor, including the  button, which turns tracking off and on. This option is the default.
 - **Always on**—Revision tracking is always on. The Review toolbar is available in LiveEditor.
 - **Always off**—Revision tracking is always off. The Review toolbar is not available in LiveEditor.
4. From the Menu bar, select **File > Save**.

Setting Up XML History Tracking for Variables

If end users can open a single Live document and edit it multiple times, you might want to track which variables changed between each version so that you can see when each end user made changes. Turning on XML history tracking creates an XML file within the DLF file structure that lets you track changes to variables for each customer. When used in conjunction with the `RevHistoryLog.xml` file, which is also part of the DLF file structure, you can see the complete revision history for a Live document.

For more information about DLF file structure, see “[A Technical Look at the DLF File Structure](#)” on page 27.

To configure XML history tracking for variables:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **XML history tracking** drop-down list, select one of the following options:
 - **None**—No changes to the variables in a DLF file are tracked.
 - **Deltas**—Only the variables that are different than the previous revision are tracked. If any elements in an array are changed, added, or removed, the entire array is written. If nothing changes, a 0-byte file is written.
 - **Full**—All of the XML data for each revision to a variable is recorded.
4. From the Menu bar, select **File > Save**.

If you select **Deltas** or **Full**, XML history files for each customer are placed in the following location in the DLF file: `DialogueLive\Data\Revision <#>`.

Specifying the Number of XML Revisions to Keep

If you selected **Deltas** or **Full** in the **XML history tracking** drop-down list, you must specify how many XML revisions will be stored in each XML history file.

To specify the number of XML revisions to keep:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. In the **XML revisions to keep** box, select the number of revisions to save.

Enter 0 in the box to keep all revisions. If you enter 1 in the **XML revisions to keep** box, no revisions are saved; only the current version is saved.
4. From the Menu bar, select **File > Save**.

18.2.2 Setting Up Revision Commenting to Capture Explanations for Changes

While revision tracking shows you what changed in a Live document, revision comments let you understand why those changes were made. Moreover, whereas you apply revision tracking and XML history tracking settings to a Live document as a whole in Design Manager, you can require revision comments to be made only on specified Live objects (in Designer) or on all Live objects (in Design Manager). As an alternative to requiring end users to comment on individual revisions, however, you can implement a revision note, which provides end users with a single place to comment on all of the changes they made to a Live document.

Keep in mind that when importing DOCX files dynamically in a Live document, any portion of text formatted as hidden in the DOCX file is converted to a comment in the Live document, including the formatting. This feature lets you preserve instructions that are already included in a DOCX document for end users who enter information. End users can search for formatting in comment text just as they can in document text.

For more information about importing DOCX files dynamically, see “[Setting Up a Live Document to Dynamically Import Text and Images at Run Time](#)” on page 60.

Note: End users will see unexpected results if they use bi-directional text (such as English and Arabic) in the same revision comment in LiveEditor.

This section discusses the following topics:

- “[Requiring End Users to Add Comments to Revisions](#)” below
- “[Prompting End Users to Enter a Revision Note](#)” on the next page

Requiring End Users to Add Comments to Revisions

Revision commenting is implemented at the object level in Designer, and at the document level in Design Manager. If you implement revision commenting at the document level, then end users are required to add comments to all revisions they make in a Live document. If you implement revision commenting at the object level, then end users are required to comment only on certain revisions.

When you require end users to add revision comments, keep in mind the following behaviors:

- If an end user makes a revision to an interactive area that is set to require revision comments (either at the object or document level), the **Create Comment** dialog box appears and the end user cannot navigate to another interactive area without entering a comment. Additionally, the **Cancel** button is not available on the **Create Comment** dialog box.
- If there are multiple changes made to an interactive area that is set to require revision comments, the comment applies to all changes made to that interactive area.
- If you have not enabled revision tracking in a Live document and an end user makes a revision to a text-based interactive area that is set to require revision comments, then none of the comments that the end user adds appear with the revision. Instead, the comments appear at the beginning of the paragraph where the revision occurred.

Requiring Revision Comments at the Object Level

1. In Designer, select the object for which you want to require revision comments.
2. Click .

The **Properties** dialog box opens for the object.

3. Click the **Interactive** tab.
4. In the **Content change** section, select the **Require revision comments** check box.
5. From the Menu bar, select **Edit > Save**.

Requiring Revision Comments at the Live Document Level

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. Select the **Enable Comments** check box.
The **Require a comment on each revision** check box becomes available.
4. Select the **Require a comment on each revision** check box.
5. From the Menu bar, select **File > Save**.

Prompting End Users to Enter a Revision Note

Revision notes make it easier for you to track the changes made to a Live document as a whole because they prompt end users to describe the changes they have made. In contrast to revision comments, which apply to individual objects in a Live document, revision notes apply to the entire Live document.

To view and edit the revision notes on the **Notes** tab, end users can select **File > Properties** on the Menu bar in LiveEditor. If end users edit notes using the **Notes** tab, they are not prompted to enter a note when they close the Live document.

To prompt end users to enter a revision note:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Prompt for revision note** drop-down list, select one of the following options:
 - **No**—Revision notes are not required. This option is the default.
 - **Prompt**—End users are prompted to enter a note when they close a Live document if they made changes.
 - **Required**—End users are required to enter a note when they close a Live document if they made changes.
4. From the Menu bar, select **File > Save**.

18.2.3 Tracking Revisions and Revision Comments in LiveEditor: The End User Experience

Because revision tracking and revision comments are available from within a Live document, this section describes how end users interact with these features in LiveEditor. In so doing, this section uses the term "you" to describe end users. All functionality described here is also detailed in the integrated help that is available in LiveEditor.

This section discusses the following topics:

- ["Customizing Revision Tracking Options" below](#)
- ["Viewing a Detailed List of Revisions" on page 276](#)
- ["Accepting and Rejecting Revisions on the Page" on page 277](#)
- ["Accepting and Rejecting Revisions in the Revision Viewer Palette" on page 278](#)
- ["Accepting All Revisions Automatically" on page 278](#)
- ["Adding Revision Comments on the Page" on page 279](#)
- ["Editing Revision Comments on the Page" on page 279](#)
- ["Adding, Editing, Viewing, and Deleting Revision Comments in the Revision Viewer Palette" on page 280](#)
- ["Viewing the Revision History in the Revision Viewer Palette" on page 282](#)
- ["Saving the Revision History in the Revision Viewer Palette" on page 283](#)

Customizing Revision Tracking Options

You can use two methods to customize revision tracking in LiveEditor: you can customize which markup is visible or you can customize the formatting of the markup (for example, you can change the color that indicates deleted text).

Selecting Which Text Revisions Are Visible

In LiveEditor, you can customize revision tracking to show the original text, new text, or both. For example, if you want to compare the original text to the new revised text, you can select **Show Original** to see the original text, and then select **Show Current** to see the revised text.

To change your revision tracking viewing options, complete one of the following tasks:

To	Do this
Make revision tracking show original text without edits	From the Tools menu, select Revision Tracking > Show Original .

To	Do this
Make revision tracking show text edits as if they were accepted	From the Tools menu, select Revision Tracking > Show Current .
Make revision tracking show text edits, but revisions are highlighted	From the Tools menu, select Revision Tracking > Show Current with Markup .

Customizing Revision Markup Format

You can also customize the way revision markups appear in LiveEditor. For example, you can change how inserted and deleted text is highlighted.

To customize the formatting of revision markup:

1. In LiveEditor, from the Menu bar, select **Tools > Revision Tracking > Markup Options**.
The **Revision Markup** dialog box opens.
2. To customize how revisions are identified on the page, you can complete one or more of the following tasks:

To	Do this
Specify which style is used on inserted text	From the Markup drop-down list, select one of the following options in the Inserted text area: <ul style="list-style-type: none">• Underline• None
Specify which color is used to highlight inserted text	<ol style="list-style-type: none">a. From the Color drop-down list, select one of the following options in the Inserted text area:<ul style="list-style-type: none">• Vary by author• Leave unchanged• Use specifiedb. If you selected the Use specified options, a color well appears next to the Color drop-down list and you can specify a color for inserted text.c. Click OK.
Specify which style is used on deleted text	From the Markup drop-down list, select one of the following options in the Deleted text area: <ul style="list-style-type: none">• Strikethrough• Hidden

To	Do this
Specify which color is used to highlight deleted text	<ol style="list-style-type: none">a. From the Color drop-down list, select one of the following options in the Deleted text area:<ul style="list-style-type: none">• Vary by author• Leave unchanged• Use specifiedb. If you selected the Use specified options, a color well appears next to the Color drop-down list and you can specify a color for inserted text.c. Click OK.

3. Click **OK**.
4. From the Menu bar, select **File > Save**.

Viewing a Detailed List of Revisions

You can review a summary of tracked changes and sort the summary of tracked changes by revision type. For example, you can sort changes based on which user made the revision. In addition, you can use the **Revisions** dialog box to accept or reject all revisions. By accepting revisions, you replace the existing text on a page with the revised text. By rejecting revisions, you keep the existing text.

To view a detailed list of revisions:

1. In LiveEditor, from the Menu bar, select **Tools > Revision Tracking > Review Changes**.
The **Revisions** dialog box opens.
2. If you want to change the sort order of the revisions, select one of the following category headings in the **Revisions** dialog box:
 - **Where**—The text box in which the change was made
 - **Change**—The type of change made
 - **Made by**—The design user who made the change
 - **Date**—The day on which the change was made
3. If you want to accept or reject revisions from within the **Revisions** dialog box, complete one of the following sets of steps:

To	Do this
Accept a revision	<ol style="list-style-type: none">Click the revision you want to accept. In the Revisions dialog box, the revision is highlighted.Click Accept. The revision no longer appears in the Revision dialog box, and the highlighting is removed from the revised text.
Reject a revision	<ol style="list-style-type: none">Click the revision you want to reject. In the Revisions dialog box, the revision is highlighted.Click Reject. The revision no longer appears in the Revision dialog box and the revisions are removed.
Globally accept or reject revisions	<ol style="list-style-type: none">Click Select All. In the Revisions dialog box, all of the revisions are highlighted.Click Accept or Reject. The revisions no longer appear in the Revision dialog box.

4. If you want to see where the revision is in the text:

- Click the revision you want to see.
- Click **Go To**.

The revision is highlighted in the text.

5. When you are finished, click **Close**.

The **Revisions** dialog box closes.

6. From the Menu bar, select **File > Save**.

Accepting and Rejecting Revisions on the Page

If you have selected either **Show Current with Markup** or **Show Current**, you can accept or reject revisions directly on a page.

To accept or reject revisions on the page, right-click the revised text and select one of the following options:

- Accept revision**—Select this option to accept an individual revision.
- Reject revision**—Select this option to reject an individual revision.
- Accept All Revisions**—Select this option to accept all of the revisions for the current customer.

- **Reject All Revisions**—Select this option to reject all of the revisions for the current customer.

Accepting and Rejecting Revisions in the Revision Viewer Palette

You might find it easier to navigate through revisions and comments using the **Revision Viewer** palette, where you can see all revisions and comments in the order they appear on each page.

You can also use the **Revision Viewer** palette to view the revision history for a particular customer or document. Each revision, after it is accepted or rejected, is recorded on a customer-by-customer basis in the revision history.

Keep in mind that you can accept and reject only individual revisions from the **Revision Viewer** palette. If you want to accept or reject all revisions, you must do so directly on the page.

To open the **Revision Viewer** palette to navigate through revisions and revision comments:

1. In LiveEditor, from the Menu bar, select **View > Revision Viewer**.
The **Revision Viewer** palette opens.
2. Click the **Current revisions** tab.
3. To accept or reject revisions, complete one of the following tasks:

To	Do this
Accept a revision	<ol style="list-style-type: none">1. Navigate to the page that contains the revision you want to accept.2. Right-click the revision in the Revision Viewer palette and select Accept revision. The revision no longer appears in the Revision Viewer palette, and the revision no longer appears on the page.
Reject a revision	<ol style="list-style-type: none">1. Navigate to the page that contains the revision you want to accept.2. Right-click the revision in the Revision Viewer palette and select Reject revision. The revision no longer appears in the Revision Viewer palette, and the revision no longer appears on the page.

4. From the Menu bar, select **File > Save**.

Accepting All Revisions Automatically

If you are using revision tracking, but only certain documents require final review before you submit them for fulfillment, you can streamline the Live document review process by using the ACCEPT_ALL LiveEditor switch. The switch lets you automatically accept all revisions and finalize any changes in documents that do not need further review. When you use the ACCEPT_ALL switch, LiveEditor performs the operation in the background and does not open the document for editing.

To automatically accept all revisions in a Live document, open the document in LiveEditor from the command prompt using the ACCEPT_ALL switch.

For example: Live.exe -ACCEPT_ALL "C:\Live Documents\MyLiveDocument.dlf"
2>AcceptAllLog.txt

For more information about the ACCEPT_ALL switch, see "[ACCEPT_ALL](#)" on page 433.

Adding Revision Comments on the Page

You can add revision comments directly to editable areas on a page in a Live document, but you must use the **Revision Viewer** palette to delete comments. Keep in mind that you cannot include formatting in a comment using the **Create Comment** dialog box, but you can subsequently edit the comment to apply formatting.

For more information about editing and formatting comments, see "[Editing Revision Comments on the Page](#)" below.

To add a comment to an editable area in a Live document:

1. In LiveEditor, right-click the object or text and select **Add comment**.
The **Create Comment** dialog box opens.
2. Enter a comment and click **OK**.
The **Create Comment** dialog box closes.
3. From the Menu bar, select **File > Save**.

Editing Revision Comments on the Page

You can edit and format existing revision comments directly within editable areas on a page in a Live document.

To edit a comment on the page in a Live document:

1. In LiveEditor, click  for an existing revision comment.
The comment opens in an editable comment box.
2. Edit the comment.
3. If you want to format the text in the comment, use the tools on the Formatting toolbar.
4. Click in the design window outside the comment box.
The comment box closes.
5. From the Menu bar, select **File > Save**.

Adding, Editing, Viewing, and Deleting Revision Comments in the Revision Viewer Palette

You can use the **Revision Viewer** palette to add, edit, view, or delete revision comments. Each end user can edit only his or her own comments; however, they can delete any comments.

To add, edit, view, and delete revision comments in the **Revision Viewer** palette:

1. In LiveEditor, from the Menu bar, select **View > Revision Viewer**.

The **Revision Viewer** palette opens.

2. Click the **Current revisions** tab.

3. To view, add, edit, or delete comments, complete one of the following sets of steps:

To	Do this
View a comment	<p>Double-click the comment that you want to view.</p> <p>The formatted comment appears in a pop-up text box on the page in the design window.</p>
Add a comment	<ol style="list-style-type: none">Go to the page where you want to add a comment.Right-click in an editable area and select Add comment. <p>The Create Comment dialog box appears.</p> <ol style="list-style-type: none">Enter a comment and click OK.
Edit and format a comment	<ol style="list-style-type: none">Double-click the comment that you want to edit. <p>The formatted comment appears in an editable comment box on the page in the design window.</p> <ol style="list-style-type: none">Edit the comment.If you want to format the text in the comment, use the tools on the Formatting toolbar.Click in the design window outside the comment box. <p>The comment box closes.</p>
Delete a comment	<ol style="list-style-type: none">Double-click the comment that you want to delete. <p>The comment appears in a comment box on the page in the design window.</p> <ol style="list-style-type: none">In the Revision Viewer palette, right-click the comment and select Delete comment. <p>A confirmation dialog box appears.</p> <ol style="list-style-type: none">Click OK.

4. From the Menu bar, select **File > Save**.

Searching for Revision Comments

To find a specific revision comment, end users can use the search functionality in LiveEditor to search text and formatting in comments just as they can search text and formatting in the Live document content.

To search for revision comments, complete the following tasks as needed:

- [“Performing Quick Searches for Revision Comments” below](#)
- [“Performing Detailed Searches for Revision Comments” below](#)

Performing Quick Searches for Revision Comments

End users can quickly search for text in revision comments in either current revisions or the revision history from the **Revision Viewer** palette.

To perform a quick search for revision comments:

1. In LiveEditor, from the Menu bar, click **View > Revision Viewer**.
2. Do one of the following:

To	Do this
Search for comments in the current revisions	Click the Current revisions tab.
Search for comments in the revision history	Click the History tab.

3. In the quick search box, type the text that you want to find.

The quick search box in the **Revision Viewer** palette



The comments list is filtered to match the search text that you entered. To search all comments or to search formatting in comments, you can click  to open the **Find** dialog box and perform a detailed search using the text that you entered.

Performing Detailed Searches for Revision Comments

End users can search for text and formatting in all revision comments using the **Find** dialog box in LiveEditor.

To perform a detailed search for revision comments:

1. In LiveEditor, from the Menu bar, click **Edit > Find**.
The **Find** dialog appears.
2. From the **Scope** drop-down list, select **Comments**.
3. Do one of the following:

To	Do this
Search for a specific word or phrase	In the Find box, enter the text that you want to find.
Search for specifically formatted text	<ol style="list-style-type: none">a. Click .b. To search for specific text, select the Text check box and enter the text you want to find.c. To search for text in a specific font, size, format, color, or style, use the corresponding check boxes, drop-down lists, and color well.

4. Click **Find**.

The first comment that matches the specified criteria appears in a comment box on the page in the design window.

Viewing the Revision History in the Revision Viewer Palette

You can use the **Revision Viewer** palette to see changes to tracked revisions in a Live document. Each revision, after it is accepted or rejected, is recorded on a customer-by-customer basis in the revision history. Because multiple people typically revise text before putting the final version into production, companies might require document revisions to be tracked for auditing purposes. For example, if your process requires approval of all edits, having a history of tracked changes makes it easier to follow each edit through the approval process.

To view the revision history of a document:

1. In LiveEditor, from the Menu bar, select **View > Revision Viewer**.
The **Revision Viewer** palette opens.
2. Click the **History** tab.

3. To filter revisions, click  and do one of the following:

To	Do this
View comments	From the Filter revisions drop-down list, select Show comments . All revision comments are visible in the Revision Viewer palette.
View revisions	From the Filter revisions drop-down list, select Show revisions . All revisions are visible in the Revision Viewer palette.
View only your revisions or comments	From the Filter revisions drop-down list, select Show my entries . Only your revisions and comments are visible in the Revision Viewer palette.
Show all revisions or comments	From the Filter revisions drop-down list, select Show all entries . Revisions and comments for all users are visible in the Revision Viewer palette.

4. From the Menu bar, select **File > Save**.

Saving the Revision History in the Revision Viewer Palette

Using the **Revision Viewer** palette, you can save the changes to tracked revisions in a customer document as XML data. The XML file that you save from the **Revision Viewer** palette is identical to the RevHistoryLog.xml file that is saved within the DLF file structure.

For more information about DLF file structure, see “[A Technical Look at the DLF File Structure](#)” on page 27.

To save the revision history of a document:

1. In LiveEditor, from the Menu bar, select **View > Revision Viewer**.

The **Revision Viewer** palette opens.

2. Click the **History** tab.

3. Click .

The **Specify the name of the XML log file** dialog box opens.

4. Browse to the appropriate location and specify a name for the file.

5. Click **Save**.

The **Specify the name of the XML log file** dialog box closes.

6. From the Menu bar, select **File > Save**.

18.3 Validating Data Entered by End Users

Since many Live documents let end users enter data, you can use variable validation to make sure that the data being entered is valid or of the correct length. For example, if you have a field that requires a telephone number, and the end user enters 40503, the data is invalid. After you have established the correct format for your data, you use the variable properties to specify the way in which end users are notified of invalid values for each variable. Accordingly, you can set different notification options for different variables. This validation functionality is frequently used with form fields.

For more information about setting up form fields, see “[Controlling End User Editing Using Selection Controls](#)” on page 76.

This section discusses the following topics:

- “[Associating a Validation Variable with an Object](#)” below
- “[Specifying the Way End Users Are Notified of Invalid Data](#)” on the next page

18.3.1 Associating a Validation Variable with an Object

Associating a validation variable with an object lets you use a variable to store the content of an object or text area. The content can then be reused by other logic (in a formula, for example) to verify that the content is valid. Depending on your selection in the settings, the end user might be required to correct the content before continuing. This functionality is available only for paragraph objects, text boxes, and text.

This topic assumes that you have already set up a validation variable.

For more information about setting up variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To associate a validation variable with an object:

1. In Design Manager, from the Library, drag the object to the Property Panel.
2. Click the **Interactive** tab.
3. In the **Variable for content and validation** box, click the  button.

The **Select Variable** dialog box opens.

4. Select a variable from the list.

5. Click **OK**.
6. From the Menu bar, select **File > Save**.

18.3.2 Specifying the Way End Users Are Notified of Invalid Data

You use variable properties to specify the way end users are notified of invalid values for each variable. (Using the **Action if invalid** drop-down list on the variable properties, you can set different notification options for different variables.) In contrast, to specify a global default way in which end users are notified of invalid values, you can use the settings on the Live setting object. Keep in mind, however, that this method overrides the option selected from the variable properties.

Tip: To create a custom message that appears when an end user enters an invalid value, use the 'SYS_ValidationMessage' variable.

For more information about specifying variable validation methods, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Validation** drop-down list, select one of the following options:
 - **Warn**—The end user receives a warning message, but can choose to continue to edit the Live document. This option is the default.
 - **Required**—The end user must correct the invalid data and cannot navigate away from the invalid field until it is corrected.

Note: The **Validation** drop-down list applies only to areas that can be edited by the end user.

4. From the Menu bar, select **File > Save**.

Chapter 19: Controlling Access to Live Documents and Areas within Live Documents

Exstream Live provides features that can help you control not only who can access a Live document during its workflow, but also which objects within a Live document can be accessed. Toward that end, you can use authentication settings on the Live document to prevent unauthorized users from accessing it, and you can control which end users have access to specific objects in the Live document by using design groups and logic. These controls allow you to support a collaborative editing experience by creating one Live document that multiple end users can edit while gaining access to only the parts of the document they have permission to change. You configure these access permissions when you design the Live document, and they are specific to each application.

This chapter discusses the following topics:

- “Controlling Access to Live Documents” below
- “Controlling Access to Areas within Live Documents” on page 290
- “Using DLF Keys to Provide Temporary Editing Access to Live Documents” on page 297

19.1 Controlling Access to Live Documents

You can restrict or permit access to Live documents by requiring users to authenticate their identity before they can open a DLF file. This process can be tied to outside authentication methods, such as LDAP, or it can use group roles created using Exstream.

To control which end users have access to a Live document, you must complete the following tasks:

1. “Defining the Authentication Settings” below
2. “Defining Design Group Access” on page 289

19.1.1 Defining the Authentication Settings

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Authorization** tab.

3. From the **User authentication method** drop-down list, select the method by which you want end users to be verified:

To	Do this
Force end users to log in using a login screen	Select Login panel .
Use end users' Windows login information to verify that end users can access the Live document (End users are not required to use a login screen in LiveEditor.)	Select ADS .
Use end users' login information for a web service to verify that they can access the Live document	Select ACL .

For more information about hosting Live documents on a web service, see “[Using a Web Service to Exchange Data Between Live Documents and Other Enterprise Systems](#)” on page 188.

4. If you selected **Login panel** as the user authentication method, select the external authentication method used from the **External authentication** method drop-down list:

To	Do this
Allow only design users created in Exstream to access the Live document	Select Disabled .
Allow Windows users and design users to access the Live document	Select Windows .

To	Do this
Use an LDAP server to authenticate the end user	<p>a. Select LDAP.</p> <p>The LDAP Authentication area appears. Use this area to identify the end user's distinguishing name and the search criteria required to find the user's design group.</p> <p>Note: Exstream Live supports clear-text passwords on an insecure channel.</p> <p>b. In the Host box, enter the name of the server that hosts the DLL server.</p> <p>c. In the Port box, enter the port on which the LDAP server is running.</p> <p>d. In the User base DSN box, enter the base distinguishing name (DN) of the location in which the LDAP server stores users.</p> <p>e. In the User RDN attribute box, enter the attribute used to identify the end user's relevant distinguishing name (RDN).</p> <p>f. In the Group search base DN box, enter the base DN of the location from which to search for groups on the LDAP server.</p> <p>g. In the Group filter box, enter the attribute that defines which end users are members of the groups. This attribute is used to identify the end user principal name which serves as the login name.</p> <p>h. In the Group attribute box, define the way groups are organized: <ul style="list-style-type: none"> • If users are listed under groups, enter DN. • If groups are listed under users, enter the attribute that identifies groups. </p> <p>i. If you want to use the LDAP server authentication in combination with the built-in Exstream user authentication, select the Include Exstream Design Users for authentication check box.</p> <p>Note: The 'SYSLD_UserGroups' system variable is automatically populated with a positive authentication. Use this variable to control access to objects within a Live document based on user permissions.</p>
Use a custom DLL to authenticate the end users	<p>a. Select External DLL.</p> <p>The External user authentication DLL area appears.</p> <p>b. In the Authentication DLL name box, click  .</p> <p>The Select the authentication DLL dialog box opens.</p> <p>c. Browse to the DLL you want to use and click Open.</p> <p>The Select the authentication DLL dialog box closes and the file name appears in the Authentication DLL name box.</p> <p>d. If you want to use the DLL authentication in combination with the built-in Exstream user authentication, select the Include Exstream Design Users for authentication check box.</p>

To	Do this
Use credentials from a web application running in Internet Explorer to authenticate end users	<p>a. Select ACL. The External URL authentication area appears.</p> <p>b. In the Authentication URL box, enter the address of the XML file that will authenticate end users.</p> <p>For more information about single sign-on authentication, see "Using Single Sign-On Authentication for Live Documents Running on a Web Service" on page 193.</p>

For more information on the other authentication methods, see *System Administration* in the Exstream Design and Production documentation.

- From the Menu bar, select **File > Save**.

19.1.2 Defining Design Group Access

To select the design groups that are authorized to use the Live document, use the **Group access** box. To set permissions and settings for each group:

- In the **Group access** box, select the design groups that are authorized to use the Live document and the permissions for each group.
- Click  .
The **Select a group to add to the list** dialog box opens.
- Select the appropriate design group and click **OK**.
The **Define group access** dialog box opens.
- In the **Access level** area, define which type of LiveEditor is available to the design group:

To	Do this
Prevent the design group from accessing the Live document in LiveEditor	Select No access .
Allow the design group to view Live documents in LiveEditor	Select Viewer .
Allow the design group to have access to all the features in the Live document, including editing privileges that you set up for objects in the file	Select Editor .

- In the **Default view** box, define the view used when a member of the design group opens the Live document. The view you select here overrides the default view specified on the **Editor Framework** tab on the setting object.

- a. Click .

The **Select View** dialog box opens.

- b. Select the view you want to use as the default and click **OK**.

The **Select View** dialog box closes and the view appears in the **Default view** box.

6. In the **Default theme** box, select the theme used when a member of the design group opens the Live document. The theme you select here overrides the default theme specified on the **Editor Framework** tab of the setting object.

- a. Click .

The **Select Theme** dialog box opens.

- b. Select the theme you want to use as the default and click **OK**.

The **Select Theme** dialog box closes and the theme appears in the **Default theme** box.

7. Click **OK**.

The **Define group access** dialog box closes.

8. From the Menu bar, select **File > Save**.

19.2 Controlling Access to Areas within Live Documents

You can allow end users to access a Live document, while at the same time specifying the parts of the document they have permission to change. Furthermore, you can customize access privileges for multiple end users of a single Live document. The following objects can be customized to grant or restrict end user access:

- Buttons
- Check boxes
- Documents
- Images
- Image selectors
- Library components
- Lines

- Pages
- Paragraphs (textual paragraphs)
- Paragraph objects
- Radio buttons
- Sections
- Signature buttons
- Shapes
- Tables
- Text boxes

This section discusses the following topics:

- “[Scenario of Multiple End Users with Different Access](#)” below
- “[Overview of the Process Used to Control Access to Objects](#)” on the next page
- “[Design Considerations for Using Design Groups to Control Editing](#)” on page 296

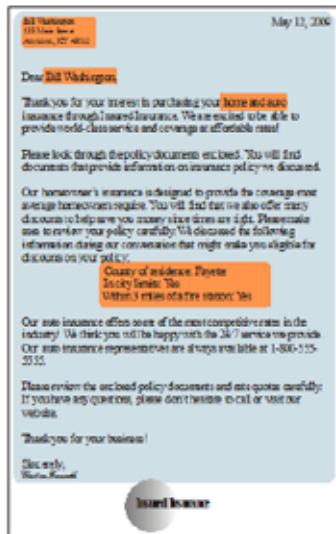
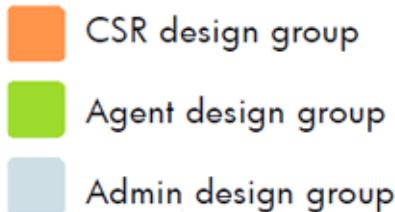
19.2.1 Scenario of Multiple End Users with Different Access

Suppose you work in the corporate office of an insurance agency, and you are designing a Live document that will allow end users to assemble insurance policies and letters for potential customers. During the editing process, three different end users must make changes to the Live document:

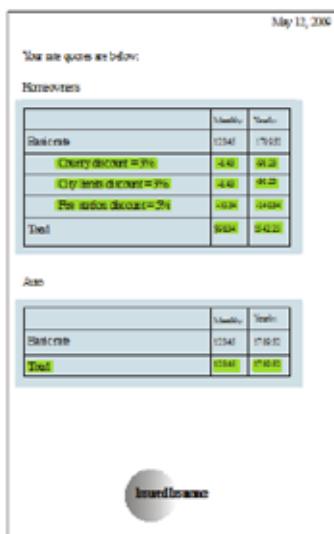
1. A customer service representative (CSR) enters the customer name and address on a cover letter while speaking to a customer on the phone and then selects the type of insurance the customer needs. The appropriate policy documents are added to the Live document automatically. CSRs cannot have access to any parts of the Live document besides the customer information text box and the policy drop-down list.
2. Next, an insurance agent views the Live document. If the customer should receive a discount (for example, a preferred rate because he or she lives in a specific county), the agent can make this change to the table that contains the rate information. The insurance agent can change only one part of the of the Live document: the tables that contain rate information.
3. Finally, the supervisor reviews the Live document before it is sent to the customer. The supervisor can change any areas on the cover letter and the rate table. However, the supervisor cannot change any of the legal information in the policy information pages.

The following graphic illustrates the areas each of these end users can edit in a single Live document.

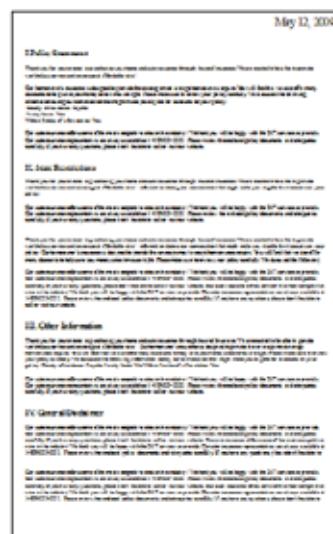
Different access based on design group



Cover Letter



Rate Tables



Legal Information

19.2.2 Overview of the Process Used to Control Access to Objects

To control end user access to Live document objects, you simply define and use traditional design objects in specific ways. Before you begin setting up the objects that control the functionality, make sure you are familiar with the design considerations associated with controlling access to objects.

For information about design considerations, see “[Getting Started Designing Interactive Documents](#)” on page [14](#).

To control end user access to Live document objects, you must complete the following tasks:

1. **Create design groups for the end users who will need to edit the Live document**
You must first determine which end users can have access to which Live document

objects. For example, some end users can access all pages in a Live document and change any of its editable parts. Other end users can access only areas that contain non-sensitive customer data. After you have determined the level of access for each end user, you create design groups based on this information. For example, if four end users need access to all pages in a Live document, you add those four users to the Admin design group.

2. **Define design group membership requirements**—After you have completed the administrative tasks of setting up design groups, you create the logic that specifies which design group a user must belong to in order to access a specific object. For example, an end user must be part of the Editor design group in order to access the Customer Information text box. This logic is contained within a variable.
3. **Set up the access requirements for each object**—Apply the variable to each object for which you want to control access. For example, to limit access to the Customer Information text box, add the formula variable specifying that the end user must belong to the Editor design group.
4. **Provide the authentication information**—Add the design groups to the setting object and identify which authentication method will be used.

When you generate the Live document, the design group information is included in the Live document and stays with it. When an end user logs in to the Live document, the login information is tested to determine which design group(s) the end user belongs to. Based on that information, any objects to which that design group does not have access are disabled in the Live document. For example, if only members of the Admin design group have access to a text box containing customer information, that text box is disabled if a member of the Editor group logs in.

This section discusses the following topics:

- “[Creating Design Groups for End Users](#)” below
- “[Defining the Group Membership Access Requirements](#)” on the next page
- “[Setting Up the Access Requirements for Each Object](#)” on page 295
- “[Providing the Authentication Information and Adding the Design Group to the Setting Object](#)” on page 295

Creating Design Groups for End Users

When you use design groups to control editing access in a Live document, you create and define users and design groups just as you would elsewhere in Exstream Design and Production.

For more information about creating and defining design users and design groups, see *System Administration* in the Exstream Design and Production documentation.

Keep the following considerations in mind as you set up design groups:

- As you add users to a design group, think about how the end users will access the Live document, and add them as the appropriate types of users to the design group. For example, if you manage end user accounts in an external repository, such as Active Directory, you can add system (network) groups to allow more flexibility in managing lists of end users.

Select the appropriate types of users and groups based on how they will access the Live document:

- Users and groups**—If the list of end users accessing the Live document does not change frequently and you can easily manage them within Design Manager, you can create and assign design users to the design group.
- System users and system groups**—If the list of end users accessing the Live document is large or changes frequently, you can specify system users and system groups that belong to the design group.
- You must assign a design user or system user for each end user who will edit the Live document. Users can belong to multiple design groups, if needed.
- Plan carefully which users will need to belong to each group. You might want to create different groups to accommodate different situations. For example, in one Live document, an end user can edit any part of the document, so that user belongs to the "DLF1Admin" group. In another Live document, however, the same end user can edit only a specific paragraph, so the user belongs to the "DLF2Edit" group.
- If you use design groups in Exstream Design and Production, you might want to implement a naming convention to help you distinguish Live design groups from Exstream design groups. For example, you can preface the names of all Live design groups with "Live_" (for example, "Live_EditGroup1").

Defining the Group Membership Access Requirements

You use variables to control which design group an end user must belong to in order to have access for a particular object. When you create the variable, you can use the `LiveTestMembership` built-in function to verify that the end user belongs to the design group you identify. The following table provides examples of a simple variable and a complex variable, both of which are used to define the group membership. The simple variable verifies membership in the Exstream Staff design group, and the complex variable verifies membership in either the Exstream Staff or the Test Group design groups.

Variable type	Example
Simple variable	<code>value = LiveTestMembership("OpenText Staff") = 0</code>

Variable type	Example
Complex variable	<pre>if (LiveEnvironment("IsAuthorizationDefined")="T") then dim levelES as integer dim levelTG as integer levelES = LiveTestMembership("OpenText Staff") levelTG = LiveTestMembership("Test Group") value-levelES = 0 or levelTG = 0 else value = True end if</pre>

You might want to consider implementing a naming convention for the variables you use to test for end user membership in design groups, since you will probably create multiple variables.

For information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

For information about the `LiveTestMembership` built-in function, see “[LiveTestMembership](#)” on page 421.

Setting Up the Access Requirements for Each Object

When you create an object to be used in a Live document, you use the **Enable** box on the **Interactive** tab of the object properties to assign the editing access level. From the **Enable** box, select the variable that defines which design groups can access each object you create. For example, if you create a text box that you want only a person of the CSR design group to access, select the variable that verifies that the end user is part of the CSR design group.

Providing the Authentication Information and Adding the Design Group to the Setting Object

Login information determines whether an end user is a member of a particular design group. Depending on which authentication method is in use, login information comes from either the login screen of the Live document or from end users' system login credentials. You must specify the authentication method on the **Authorization** tab of the Live setting object to enforce editing permissions using design groups.

When an end user logs in to the Live document, the end user's login information is evaluated to determine whether the end user is part of a design group. Then, all of the areas available to that design group are made available in the Live document. Since the information about which design groups have edit permissions is specific to each Live document, this information stays with the Live document. You must provide a list of the design groups that have editing access in the **Group Access** box on the **Authorization** tab of the Live setting object.

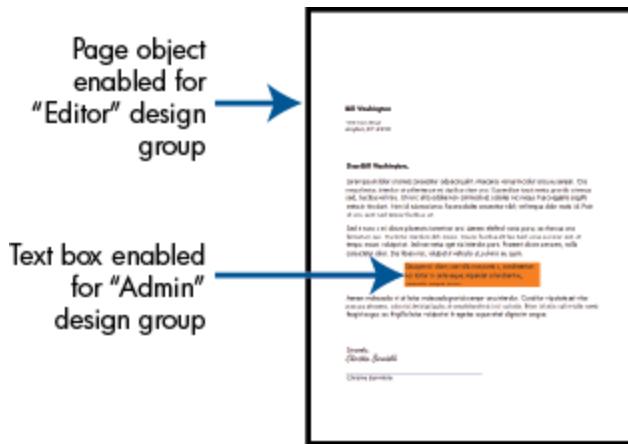
For information about defining design group access, see “[Defining Design Group Access](#)” on page 289.

19.2.3 Design Considerations for Using Design Groups to Control Editing

When you set up a Live document in which you want to use design groups to restrict editing, you must make sure that end users are assigned to the correct design groups so they will have access to the content they want to change. You must also plan your design carefully to ensure you can restrict the document as necessary. Keep the following design considerations in mind:

- If an object is within another object, the design group must have edit access to the larger object in order to access the contained object. For example, if members of the Editor group want to access a paragraph within a text box, they must have access to both the text box and the paragraph.
- Although a variable can be located in a text paragraph, it is not considered to be contained by the paragraph object. Therefore, if you want to allow end users to edit variable values in a paragraph, you must enable editing for each individual variable, rather than enable editing for the overall paragraph object. Even if you do not enable editing for the paragraph, the variables in the paragraph will be editable for each end user with permission to edit them.
- If editing for an embedded object is enabled for a different design group than its containing object, then only those end users authorized to edit the embedded object can edit that object. In other words, end users must belong to both design groups in order to access the inner object. The following example illustrates this concept:

Access behaviors



The following table demonstrates how end users who are part of various design groups can interact with the page and text box objects:

Design group interaction with Live document objects

If the end user belongs to design group(s)	End user can edit
Admin and Editor	Page and text box
Editor	Page only (not the text box)
Admin	Text box

You can use two methods to enable an object for more than one design group:

- Set up the design groups so end users in one design group are always members of another design group. You can accomplish this using either of the following methods:
 - Add an entire design group to another design group. For example, you can add the Admin design group to the Editor design group. In this case, all members of the Admin design group automatically have all the same access permissions as the Editor group. In the design group properties, use the **Groups included in group** area to add a design group to another design group.
 - Add all the individual users in one design group to an additional design group. For example, you can add all the members of the Admin design group to the Editor design group.
- Provide logic in the variable that checks whether an end user is a member of at least one of multiple design groups.

19.3 Using DLF Keys to Provide Temporary Editing Access to Live Documents

If you have licensed the Live Certificates module, you can use DLF keys (also called DLF certificates or Live certificates) to allow end users to edit Live documents in LiveViewer, and to authorize optional modules for use within LiveViewer. Because DLF keys allow limited editing functionality in the free LiveViewer, they let you distribute your Live documents to a wider audience.

Additionally, DLF keys let you set a date range for when a Live document is valid, and specify which actions are allowed (in both LiveViewer and LiveEditor) after the Live document has expired. For example, if you distribute an enrollment form that end users must complete by a certain date, you can include a DLF key in the Live document that prevents end users from editing the enrollment form after the deadline.

This section discusses the following topics:

- “[Types of DLF Keys](#)” below
- “[Setting Up DLF Keys](#)” on the next page
- “[Applying DLF Keys to Live Documents](#)” on page 301

19.3.1 Types of DLF Keys

You can include two kinds of DLF keys in a Live document: a non-editable DLF key based on a DLF key file (a file with the .dkf extension) and a cloned, editable DLF key.

DLF Keys Based on DKF

When applying DLF keys to a Live document, you must begin with a DLF key based on a DLF key file (.dkf file). A .dkf file is similar to the Exstream License key (which comes from a file with the .ekf extension) that you load into the **System Settings**. However, the DLF key controls only options related to editing a Live document. A DLF key file is provided when you license the Live Certificates module. DLF keys based on a .dkf file are not editable.

DLF keys based on a .dkf file show you the following information:

- How long your .dkf file is valid
- Which modules are available
- Default expiration options and messages

Cloned DLF Keys

If you want to make changes for a particular customer or project, such as a change in the expiration date, you can create a clone of a DLF key. Clones of DLF keys are editable. Therefore, you can create custom DLF keys that are subsets of your DLF key and send them to customers. You can limit the amount of time that customers can use a Live document or what customers can do with the Live document after they are finished editing, such as printing and saving.

Clones of DLF keys are always clones of a DLF key based a .dkf file. For example, suppose you use a DLF key based on a .dkf file called C and a clone called C2. If you create a clone of C2 called C3, C3 is actually a clone of C. C3 does not take any of its properties from C2.

19.3.2 Setting Up DLF Keys

This section discusses the following topics:

- “[Creating a DLF Key Based on a .dkf File](#)” below
- “[Creating an Editable DLF Key](#)” below

Creating a DLF Key Based on a .dkf File

1. In Design Manager, in the Library, go to **Environment > Interactive > DLF Keys**.
2. Right-click the **DLF Keys** heading and select **New DLF Key**.

The **New DLF Key** dialog box opens.

3. In the **Name** box, enter a name.
4. In the **Description** box, enter a description (optional).
5. In the **DLF Key file (.dkf)** box, click  to browse to the .dkf file.
6. Click **OK**.

The properties for the new DLF key open in the Property Panel.

7. From the Menu bar, select **File > Save**.

Creating an Editable DLF Key

To create an editable DLF key, right-click the DLF key on which you want to base a new DLF key and select **Clone**. Give the new DLF key a unique name.

Clones of DLF keys are always clones of a DLF key based on a .dkf file. For example, suppose you use a DLF key based on a .dkf file called C and a clone called C2. If you create a clone of C2 called C3, C3 is actually a clone of C. C3 does not take any of its properties from C2.

To create an editable DLF key:

1. In Design Manager, in the Library, go to **Environment > Interactive > DLF Keys**.
2. Right-click the **DLF Keys** heading and select **New DLF Key**.

The **New DLF Key** dialog box opens.

3. In the **Name** box, enter a name.
4. In the **Description** box, enter a description (optional).
5. In the **DLF Key file (.dkf)** box, click  to browse to the .dkf file.

6. Click **OK**.

The properties for the new DLF key open in the Property Panel.

7. Adjust the properties as necessary:

To	Do this
Permit end users to use the Live document as part of a post-processing routine or as data in the engine	<p>In the Customer ID authorized for engine processing text box, add the appropriate customer IDs. This option limits the users who can use the Live document as data. If you leave the Customer ID authorized for engine processing box empty, anyone can use the Live document as data.</p> <p>For example, suppose you create a survey and want to keep the information anonymous and secure. You send the Live documents to a third party for data extraction, and include the third party's customer ID to prevent others from using the data and passing it into Exstream.</p>
Specify how long the DLF key is valid	<p>In the Valid time period section, adjust the dates as necessary.</p> <p>In the Grace period (days) box, select the number of days after the expiration date that a Live document can still be edited. If there is no grace period, the Live document becomes non-editable after the DLF key expires. If there is a grace period, a warning will be issued in LiveViewer after the DLF key expires, but the document can continue to be edited until the grace period expires. After the grace period expires, the Live document can be viewed but not edited in LiveViewer.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"><p>Note: The engine can process a Live document with an expired key in a fulfillment process.</p></div>
Allow end users to save Live documents as PDFs	<p>In the Authorized DLF Modules area, select the Save as PDF check box.</p> <p>Depending on your Live implementation, the Save as PDF check box might not be available.</p>

To	Do this
Specify the actions end users can complete outside of the valid date range (before the start date and after the end of the grace period)	<p>In the Expiration options section, adjust the settings as necessary:</p> <ul style="list-style-type: none"> • Allow opening—Select this option to allow end users to open an expired Live document. • Allow printing—Select this option to allow end users to print an expired Live document. • Allow "save as"—Select this option to allow end users to save an expired Live document using a different name. • Allow "save as PDF"—Select this option to allow end users to save an expired Live document as a PDF. • Expiration options apply to LiveEditor—Select this option to override the preferences set in settings and views. By default, the greater functionality is available to end users. For example, if you allow printing in LiveEditor but clear the Allow printing check box, end users can still print unless you select the Expiration options apply to LiveEditor check box. • Expiration warning message—During the grace period, if end users attempt an action not allowed in the Expiration options area, they receive a warning message from LiveViewer stating that the Live document has expired and that they are still allowed to perform the action for a limited grace period. If you want to provide additional information, such as contact information or the features that end users can still access after the end of the grace period, enter text in the Expiration warning message box. • Expiration message—End users receive expiration messages from LiveEditor when they attempt an action that is not allowed according to the selections in the Expiration options and is outside of the valid date range (before the start date and after the expiration plus any grace period). End users also receive a message when they attempt to use a module they are not authorized to use. <p>Use the Expiration message box to enter any information, in addition to the default, that you want to provide end users. For example, you can provide contact information, such as a phone number, that end users can use if they have questions. By default, end users are notified of valid dates if they attempt an action before the start date; they are notified of the date the Live document expired if they attempt an action after the end of the grace period; and they are notified that a module is not authorized for use if they attempt to use a feature in an unauthorized module.</p> <p>Keep in mind that the availability of options in the Expiration options area is affected by the modules that are available. For example, if the Save as PDF check box is cleared, the Allow "save as PDF" check box is inactive.</p>

8. From the Menu bar, select **File > Save**.

19.3.3 Applying DLF Keys to Live Documents

After you create a DLF key, you add it to a Live setting object. You then add the setting object to a DLF output object. When you create the Live document output, the setting (and therefore the DLF key) is applied.

To add a DLF key to a setting object:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Authorization** tab.
3. In the **DLF Key** box, select the DLF key you want to associate with the setting object.

- a. Click .

The **Select DLF Key** dialog box opens.

- b. Select the DLF key you want to use and click **OK**.

The **Select DLF Key** dialog box closes and the key appears in the **DLF Key** box.

4. To let end users edit a Live Document with the Viewer version of LiveEditor, select the **Allow editing in LiveViewer** check box.
5. From the Menu bar, select **File > Save**.

Chapter 20: Implementing Security Features in Live Documents

Exstream Live provides several security features to protect the data and content in Live documents from being viewed or edited by unauthorized users. You can encrypt DLF files throughout your workflow to make sure that only authorized users can access the files. You can also prevent content within the Live document from being displayed in the thumbnail previews of DLF files so that sensitive data is not displayed. In addition, you can use privacy masks to prevent unauthorized users from viewing sensitive data during the editing process. Finally, you can include signature buttons in Live documents that allow certain end users to electronically sign and lock the documents to prevent further changes.

This chapter discusses the following topics:

- “[Encrypting Data and Content in a Live Document](#)” below
- “[Preventing Content from Appearing in Thumbnail Previews of Live Document Files](#)” on page 307
- “[Protecting Variable Data by Using Privacy Masks](#)” on page 308
- “[Using Signature Buttons to Allow Certain End Users to Prevent Changes in Live Documents](#)” on page 311

20.1 Encrypting Data and Content in a Live Document

If a Live document contains sensitive data, you can encrypt some or all of the XML content stored in the DLF file to prevent unauthorized users from viewing the information outside of LiveEditor. The Live document appears normally in LiveEditor, but the data cannot be viewed by exploring the DLF file in a file explorer program.

Data in DLF files is encrypted using the RC4 stream cipher with a 56-bit key.

If you want to allow external systems to access an encrypted DLF file, you must run the engine separately from the Live document fulfillment step, using Live encryption switches.

For more information about engine switches for Live encryption, see “[Live Encryption Switches](#)” on the next page “[Live Encryption Switches](#)” on page 442.

To encrypt the data in the Live document:

1. In Design Manager, from the Library, drag the output object associated with the Live document to the Property Panel.

2. Click the **Security** tab.
3. Use the check boxes on the **Security** tab to encrypt the data that you want to protect:

To	Do this
Encrypt the XML content that is related to the data stored for each customer	Select the Encrypt customer data check box.
Encrypt the XML content that is related to the content stored within the DLF file	Select the Encrypt content/manifest check box.
Encrypt the XML content that is related to the metadata about the DLF file	Select the Encrypt metadata check box.
Encrypt the data that is included on interview pages	<p>Select the Encrypt interview XML data check box.</p> <p>You might want to make this selection if you are presenting the Live document as a web form.</p> <p>For information about creating Live documents that can be presented as web forms, see "Collecting Data For a Live Document Using Web Forms" on page 197.</p>

4. From the Menu bar, select **File > Save**.

20.1.1 Live Encryption Switches

The switches discussed in this section can be used from the command prompt or from a control file to encrypt or decrypt XML content within Live documents. The switches discussed here are specific to the Live environment, and they are used separately from any processing switches.

For information about encrypting Live documents, see ["Encrypting Data and Content in a Live Document" on the previous page](#).

When using Live encryption switches, you must also use the KEY switch, KEYFILE switch, or KEYPART switch to specify a key.

For more information about using switches to specify a key, see *Preparing Applications for Production* in the Exstream Design and Production documentation.

This section discusses the following switches that are used to manage Live document encryption:

- ["DLF_CIPHER" on the next page](#)
- ["DLF_CIPHERLIST" on the next page](#)
- ["DLF_CIPHERLOG" on page 306](#)
- ["DLF_CIPHEROPTIONS" on page 306](#)
- ["DLF_CIPHEROUT" on page 307](#)

DLF_CIPHER

Use the DLF_CIPHER switch to encrypt or decrypt a single DLF file.

Use the following arguments with the DLF_CIPHER switch:

Argument name	Use	Description
Mode	Required	This argument specifies whether the Live document should be encrypted or decrypted. Use one of the following options: <ul style="list-style-type: none">• ENCRYPT• DECRYPT
FileLocation	Required	The location of the DLF file

For example:

```
-DLF_CIPHER=DECRYPT,C:\DLFs\LiveDocument.dlf
```

DLF_CIPHERLIST

Use the DLF_CIPHERLIST switch to encrypt or decrypt multiple DLF files listed in a separate text file. The file specified by the DLF_CIPHERLIST switch must contain the absolute or relative path to each DLF file to be encrypted or decrypted (with one path on each line).

Sample file containing list of DLF files

```
C:\Program Files\Exstream\CustomerWelcome003423.DLF
C:\Program Files\Exstream\CustomerWelcome003553.DLF
C:\Program Files\Exstream\CustomerWelcome003801.DLF
C:\Program Files\Exstream\CustomerWelcome004463.DLF
C:\Program Files\Exstream\CustomerWelcome004493.DLF
C:\Program Files\Exstream\CustomerWelcome004528.DLF
C:\Program Files\Exstream\CustomerWelcome004687.DLF
C:\Program Files\Exstream\CustomerWelcome005923.DLF
C:\Program Files\Exstream\CustomerWelcome006702.DLF
C:\Program Files\Exstream\CustomerWelcome006723.DLF
```

Use the following arguments with the DLF_CIPHERLIST switch:

Argument name	Use	Description
Mode	Required	This argument specifies whether the Live documents should be encrypted or decrypted. Use one of the following options: <ul style="list-style-type: none">• ENCRYPT• DECRYPT

Argument name	Use	Description
ListFileLocation	Required	The location of a text file that contains a list of DLF file locations

For example:

```
-DLF_CIPHERLIST=DECRYPT, C:\Program Files\OpenText\CustomerWelcomeList.txt
```

If you are working in a Multiple Virtual Storage (MVS) environment, you cannot use the DLF_CIPHEROUT switch to specify a different output location for multiple DLF files listed in a file specified by the DLF_CIPHERLIST switch, and the engine will overwrite the specified DLF files with the encrypted or decrypted versions.

DLF_CIPHERLOG

Use the DLF_CIPHERLOG switch to specify a location for the log of the encryption or decryption event. If you do not use this switch, the log file is written to the same folder as the engine.

The logged items are as follows:

- Engine version
- Date
- Mode (ENCRYPT or DECRYPT)
- Each file encrypted or decrypted and whether the action is completed successfully

The only argument is the log file location.

For example:

```
-DLF_CIPHERLOG=C:\logs\CipherLog.txt
```

DLF_CIPHEROPTIONS

Use the DLF_CIPHEROPTIONS switch to specify which data in the Live document should be encrypted or decrypted. If you do not use this switch, the action applies to all XML content within the Live document.

The only argument is the content to be processed. You can specify multiple options. Use one of the following options:

- CUSTOMER—The XML content related to the data stored for each customer
- CONTENTMANIFEST—The XML content related to the content stored within the Live document
- METADATA—The XML content related to the metadata about the Live document
- INTERVIEW—The XML content included on interview pages
- ALL—All XML content within the Live document

For example:

```
-DLF_CIPHEROPTIONS=CUSTOMER,METADATA
```

DLF_CIPHEROUT

Use the DLF_CIPHEROUT switch to specify a location for the encrypted or decrypted output file. If this switch is omitted, the engine overwrites the original file with the encrypted or decrypted version. Keep in mind that if you are processing multiple DLF files, only the file path specified is used, and any file name is ignored.

The only argument is the file location.

For example:

```
-DLF_CIPHEROUT=C:\DLFs\DecryptedFile.DLF
```

If you are working in a Multiple Virtual Storage (MVS) environment, you can use the DLF_CIPHEROUT switch only when you encrypt or decrypt a single Live document using the DLF_CIPHER switch. If you specify multiple Live documents using the DLF_CIPHERLIST switch, you cannot use the DLF_CIPHEROUT switch, and the engine will overwrite the specified DLF files with the encrypted or decrypted versions.

20.2 Preventing Content from Appearing in Thumbnail Previews of Live Document Files

By default, end users can view a thumbnail image of a DLF file anywhere Windows or another operating system allows thumbnail viewing. For example, if end users open Live documents using the **File** menu, they can see a thumbnail of the Live documents in the **Open** dialog box by changing the view to thumbnail view. You can prevent preview thumbnail images of Live documents from appearing to help protect sensitive data in the files.

To prevent Live document content from appearing in thumbnails:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Authorization** tab.
3. Select the **Hide preview thumbnails** check box.
4. From the Menu bar, select **File > Save**.

20.3 Protecting Variable Data by Using Privacy Masks

Since variables can be used to contain sensitive customer data, such as credit card numbers or account numbers, you often need to protect the data from being viewed by unauthorized users during the editing process. For example, if a Live document will be edited by four different people before it is processed, only one of those people might be authorized to see customer Social Security numbers. You can apply privacy masks to variables in Design Manager to prevent variable data from being visible in LiveEditor. When you use privacy masks, characters that you specify appear in place of the data. For example, instead of the account number 1111-2222-3333-4444, the data can appear as XXXX-XXXX-XXXX-4444. When you define a privacy mask for a variable, the privacy mask appears in every location of the variable in the Live document.

Privacy masking is different from the confidentiality feature you can use when you map data. The confidentiality feature is used to specify that a data area contains confidential information, but the data still appears in the output.

For information about using the confidentiality feature, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

Privacy masking is also different from the **Default data entry mask** property on the **Interactive** tab of the variable properties. The data entry mask allows you to control the appearance of data that end users enter in an editable variable. When you set up a privacy mask, you use the default data entry mask information to build the privacy mask. However, you are not required to use a privacy mask in order to control how end users enter data in form fields.

For information about the **Default data entry mask** property, see “[Using Form Fields to Control Data Entry](#)” on page 113.

For more information about setting up and using variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

This section discusses the following topics:

- “[About Privacy Masks and End User Editing](#)” below
- “[Using Privacy Masks with Output Formatting](#)” on the next page
- “[About Privacy Masks and Live Document Data](#)” on the next page
- “[Setting Up Privacy Masks](#)” on page 310

20.3.1 About Privacy Masks and End User Editing

Your workflow and security standards might require that some end users be allowed to see sensitive data values, while some end users are not. For example, you might want to allow end

users at a supervisor level to see and edit real data values, while all other end users see only the privacy mask. You can use the `LiveSetPrivacyMask` built-in function, in conjunction with user group information, to "turn on" or "turn off" privacy masks when different types of end users open the Live document.

If a variable that uses a privacy mask is editable, then when an end user clicks the variable to edit its value, the privacy mask is disabled for the new entry so that the end user can see the changes. When the end user selects a different object, the privacy mask is enabled again.

Any sensitive data that appears for an end user in LiveEditor instead of a privacy mask is also visible when the end user prints the Live document or saves it as a PDF. However, if only a privacy mask appears in LiveEditor, the privacy mask is also used when the end user prints the Live document or saves it as a PDF, and the sensitive data is not visible.

For information about using user information to control how the Live environment appears to different end users, see ["Controlling Access to Live Documents and Areas within Live Documents" on page 286](#).

For information about the `LiveSetPrivacyMask` built-in function, see ["LiveSetPrivacyMask" on page 416](#).

20.3.2 Using Privacy Masks with Output Formatting

When you use the **Output Format** area on the **Output Format** tab of the variable properties to apply output formatting to a variable, the **Special Formatting** setting in the variable properties in Designer inherits this formatting. When the engine runs, it applies the formatting specified in the **Special Formatting** property to the variable. Then, any formatting designated by the privacy mask is subsequently applied when the Live document is opened in LiveEditor. The privacy mask formatting does not override output formatting; instead, it is added to the output formatting. For example, if you apply an output format for a date variable that is set up to be `##/##/####`, then when the engine runs, the date appears as `12/12/2020`. However, if you also set up a privacy mask for the variable that is designed with the same structure, then when the Live document is opened, the date will incorrectly appear as `12//1/20`.

Therefore, if you want to use a privacy mask and output formatting on the same variable, make sure that the two settings will not conflict with each other. For example, if you want the final date variable appearance to be `12/12/2020`, you can set the output format to be `mmddyyyy`. Then, when you apply the privacy mask (`##/##/####`), the variable will appear correctly in LiveEditor.

For more information about setting up an output format, see [Using Data to Drive an Application](#) in the Exstream Design and Production documentation.

20.3.3 About Privacy Masks and Live Document Data

When you use a privacy mask to protect data, the mask affects only the visual appearance of the data in LiveEditor. In the underlying customer XML of the DLF file, the data values do not

change. Therefore, you can continue to use the customer data in other processes, or you can archive it as part of your records management. However, this also means that variable data is visible to users who view the customer XML. If you need to protect the data in the customer XML, you must encrypt the Live document using the encryption options on the Live output object.

For information about Live document encryption, see [“Encrypting Data and Content in a Live Document” on page 303](#).

20.3.4 Setting Up Privacy Masks

1. In Design Manager, from the Library, drag the variable you want to mask to the Property Panel.
2. Click the **Interactive** tab.
3. Use the privacy mask boxes to define how the privacy mask works with the variable:

To	Do this
Mask all the characters in the variable value, regardless of the length and format of the data value (for example, if the value can be anything from 1,000 to 100,000)	<ol style="list-style-type: none">a. In the Mask character box, enter the character you want to use to mask the data.b. Select the Mask all characters with the specified Mask character without supplying a data entry mask check box.
Mask certain characters in the variable value or mask all the characters while honoring the format specified by the default data entry mask	<ol style="list-style-type: none">a. In the Default data entry mask box, enter a data entry mask. You will use this information to build the privacy mask. The default data entry mask information also appears in the Privacy mask box. For information about defining a default data entry mask, see “Using Form Fields to Control Data Entry” on page 113.b. In the Mask character box, enter the character you want to use to mask the data. For example, if you are applying a privacy mask to a 'CreditCardNumber' variable, you might use the X character.c. If you want to apply the privacy mask character to all of the characters in the data, click Mask all. To apply the privacy mask character to specific characters, continue to step d.d. In the Privacy mask box, click each character to which you want to apply the privacy mask. For example, if you are applying a privacy mask to a 'CreditCardNumber' variable, you might choose to click the first 12 numbers in the data field and leave the last four numbers unmasked. The mask characters in the Privacy mask box are updated to reflect the mask you applied. When the Live document is produced as output, the data in the variable will use the mask you specify. For example, the 'CreditCardNumber' variable might appear as XXXX-XXXX-XXXX-8432 in the Live document.
Remove the privacy mask	Click Unmask all .

4. From the Menu bar, select **File > Save**.

20.4 Using Signature Buttons to Allow Certain End Users to Prevent Changes in Live Documents

You can use a signature button in a Live document to let an end user add an electronic signature in order to lock all or part of the Live document to prevent further changes. For example, if you send the Live document to an approver after the initial changes have been made, you can set up a signature button that the approver clicks to lock the document after he or she has approved it.

If you want to use a signature device or other method, such as a signature pad, with a signature button, you must install or configure the device or method before setting up the signature button.

Electronic signatures used in Live documents are different from electronic signatures used in traditional Exstream Design and Production applications. Electronic signatures used in traditional Exstream Design and Production applications allow you to integrate a document with an electronic signature solution, such as a cloud-based offering from an electronic signature vendor.

For information about using electronic signatures in traditional Exstream Design and Production applications, see *Designing Customer Communications* in the Exstream Design and Production documentation.

To set up a signature button, you must complete the following tasks:

1. [“Adding a Signature Button to a Page” below](#)
2. [“Defining the Variable Settings of the Signature Button” on the next page](#)
3. [“Customizing the Appearance of the Signature Button” on page 313](#)
4. [“Selecting the Scope of the Signature” on page 315](#)

You can also complete the following optional tasks as needed:

- [“Specifying Additional Signature Button Actions” on page 316](#)
- [“Preventing End Users From Removing a Signature” on page 317](#)

20.4.1 Adding a Signature Button to a Page

In Designer, on the Drawing Objects toolbar, click . A signature button is placed on the page. You can use the tools in Designer to move and resize the button as needed.

20.4.2 Defining the Variable Settings of the Signature Button

You use the options in the variable settings area of the signature button properties to specify the variable to be populated with a value when the signature button is selected, and how the value of that variable is stored.

Before beginning this task, you must have created a variable designed to store the value of the signature button when it is selected.

For more information about creating variables, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

To define the variable settings of the signature button:

1. In Designer, select the signature button and click .

The **Signature Button Properties** dialog box opens.

2. Click the **Button** tab.
3. Use the **Variable** box to specify the variable that will be populated with a value when an end user selects the signature button. The variable must be a Boolean, string, or integer variable.

- a. Click .

The **Select Variable** dialog box opens.

- b. Select the variable and click **OK**.

The **Select Variable** dialog box closes and the variable you selected appears in the **Variable** box.

4. If you specified an array variable in the **Variable** box, then in the **Array element** box, enter the index value of the array element that will be populated when an end user selects the signature button. Enter **0** if you want the first empty element to be populated automatically.
5. To specify the value stored when the signature button is selected, complete one of the following sets of steps:

To	Do this
Store a static specified value	In the Value if clicked box, enter the value you want to assign to the specified variable when the signature button is selected. This box is active only if you specified a string or integer variable.

To	Do this
Store a variable-driven value	<p>a. In the Array containing selection values box, click  . The Select Variable dialog box opens.</p> <p>b. Select the variable that provides the values and click OK.</p> <p>The Select Variable dialog box closes and the variable you selected appears in the Array containing selection values box.</p>

6. From the Menu bar, select **Edit > Save**.

If you use an array variable to store the value of the signature button, and the signature button is placed in an object that can repeat dynamically (such as a table row), then the elements in the array are populated with values in the order that the signature buttons appear. For example, if the signature button is located in a repeating table row, the first element in the array contains the value for the signature button in the first row, the second element contains the value for the signature button in the second row, and so on.

If you use array variables to set up the signature button values, and the signature button is placed in an automated data-section driven table, all of the variables used to define the signature button must be associated with the data section. Therefore, if you cannot map the variable that stores the signature button value in the data section, you must use the VARSCOPE switch in a control file to specify the section that is associated with the variable.

For more information about the VARSCOPE switch, see *Switch Reference* in the Exstream Design and Production documentation.

20.4.3 Customizing the Appearance of the Signature Button

1. In Designer, select the signature button and click  .
The **Signature Button Properties** dialog box opens.
2. Click the **Button** tab.

3. Use the **Button style** option and complete the remaining steps to specify the appearance of the signature button:

To	Example	Do this
Display a standard button with custom text		<ul style="list-style-type: none"> a. From the Button style drop-down list, select Drawn (button). b. In the Caption box, enter the text you want to appear on the button before an end user clicks it. For example, you might enter Sign. c. In the Down caption box, enter the text that appears on the button after an end user clicks it. For example you might enter Signed.
Display a box that changes the caption when an end user selects it		<ul style="list-style-type: none"> a. From the Button style drop-down list, select Drawn (box). b. In the Caption box, enter the text you want to appear in the box before an end user selects it. For example, you might enter Sign. c. In the Down caption box, enter the text that appears in the box after an end user selects it. For example you might enter Signed.
Display a box that is replaced with a digital signature image when an end user selects it	Varies	<p>Before setting up a box that is replaced with a digital signature image, you must have created a variable with a Type of Placeholder and a Placeholder Type that is an image file format.</p> <ul style="list-style-type: none"> a. From the Button style drop-down list, select Drawn (box) / Image. b. In the Caption box, enter the text you want to appear in the box before an end user selects it. c. In the Down image placeholder variable box, click <p>The Select Variable dialog box opens.</p> <ul style="list-style-type: none"> d. Select the placeholder variable that provides the signature image that you want to appear after the end user selects the signature box, and click OK. <p>The Select Variable dialog box closes and the variable you selected appears in the Down image placeholder variable box.</p> <p>The digital signature image data is stored within the Live document to be used by the engine if the Live document is imported at run time (for example, if the DLF file is imported into a PDF).</p>

To	Example	Do this
Display a specified image that changes when an end user selects it	Varies	<ol style="list-style-type: none">a. From the Button style drop-down list, select Images. Additional options appear at the bottom of the dialog box.b. Double-click the Up box. The Import an Image dialog box opens.c. Browse to and select the image that you want to be used for the button before an end user selects it.d. Click Open. The Import an Image dialog box closes and the image you selected appears in the Up box.e. Repeat step b through step d for the Down box (the image used after the end user selects the button) and Hover box (the image used when an end user hovers the pointer over the button).

4. Click **OK**.

The **Signature Button Properties** dialog box closes.

5. From the Menu bar, select **Edit > Save**.

After you have defined the basic signature button properties, you can use some of the other properties in Designer to customize its appearance and behavior. For example, to change the color and border of the signature button, you can use the options on the **Lines and Fill** tab of the **Signature Button Properties** dialog box.

Tip: If you want to let end users sign multiple locations at once, save the signature button as a Library component. You can then populate all the signatures from a single location.

For more information about Designer object properties and creating Library components, see *Designing Customer Communications* in the Exstream Design and Production documentation.

20.4.4 Selecting the Scope of the Signature

To select how much of the Live document is locked after an end user adds an electronic signature:

1. In Designer, select the signature button and click .

The **Signature Button Properties** dialog box opens.

2. Click the **Button** tab.

3. From the **Signature scope** drop-down list, select one of the following options:

To	Do this
Lock the entire Live document	Select File .
Lock the active customer	Select Customer .
Lock the active document within the Live document	Select Document .
Lock the active page	Select Page .
Not lock anything in the Live document	Select None .

4. From the Menu bar, select **Edit > Save**.

End users can still print, save, save as, add additional signatures to, remove the signature from, and open locked Live documents, but they cannot change any content.

20.4.5 Specifying Additional Signature Button Actions

When an end user clicks a signature button, a function is used to complete any additional actions associated with signing the Live document.

Before beginning this task, you must have created a function designed to carry out the intended action of the signature button.

For information about creating functions, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

For information about Live built-in functions, see “[Live Built-In Functions](#)” on page 381.

To specify the signature button actions:

1. In Designer, select the button and click . The **Button Properties** dialog box opens.
2. Click the **Interactive** tab.
3. Click . The **Advanced properties** dialog box opens.
4. In the **Button action** box, click . The **Select Function** dialog box opens.
5. Select the function that provides the action you want to launch when the button is clicked

and click **OK**.

The **Select Function** dialog box closes and the function you selected appears in the **Button action** box.

6. Click **OK**.

The **Advanced properties** dialog box closes.

7. Click **OK**.

The **Button Properties** dialog box closes.

8. From the Menu bar, select **Edit > Save**.

20.4.6 Preventing End Users From Removing a Signature

To prevent end users from removing an electronic signature and unlocking a Live document for further editing after it has been signed:

1. In Designer, select the signature button and click .

The **Signature Button Properties** dialog box opens.

2. Click the **Button** tab.

3. Select the **Locked on sign** check box.

4. From the Menu bar, select **Edit > Save**.

Chapter 21: Testing Live Documents before Moving into Production

An important part of creating a new application is troubleshooting and testing it. In addition to performing traditional application testing on a Live document, you can also use specific tools available with Exstream Live to double-check and validate the interactive parts of a Live document. Exstream Live provides several tools you can use to test the various components of a Live document, both during the design phase and before the document is finalized.

For information about the basic testing tools available through Exstream, see *Preparing Applications for Production* in the Exstream Design and Production documentation.

You can use the following types of tools to test a Live document:

Testing tools available with Live documents

Testing task	Description of test tool
Previewing the interactive and other Live features in a Live document	The Live Preview mode in Designer lets you interact with most parts of a Live document, just as an end user would. Since you use Live Preview in Designer, you can easily toggle back and forth between Live Preview mode and the design mode to make and test changes.
Previewing output colors in a Live document	The Preview Output Colors functionality in Designer and LiveEditor lets you sample colors from design pages or Live documents and replace them with corresponding colors from CMYK output files. You can use this feature to compensate for the differences inherent between the RGB and CMYK color models.
Finding all Live objects in a Live document	A special search mode makes it easy to locate all the objects on a specific page that have Live properties. You can use this feature to help troubleshoot Live objects or to easily locate and make adjustments to the Live objects in a design.
Testing the logic in a Live document	A special tool called the Debugger is available in LiveEditor to help you test the logic and dynamic components in a Live document. This tool allows you to watch the behavior of specific objects in your design and easily identify the adjustments you must make to your written logic.

This chapter discusses the following topics:

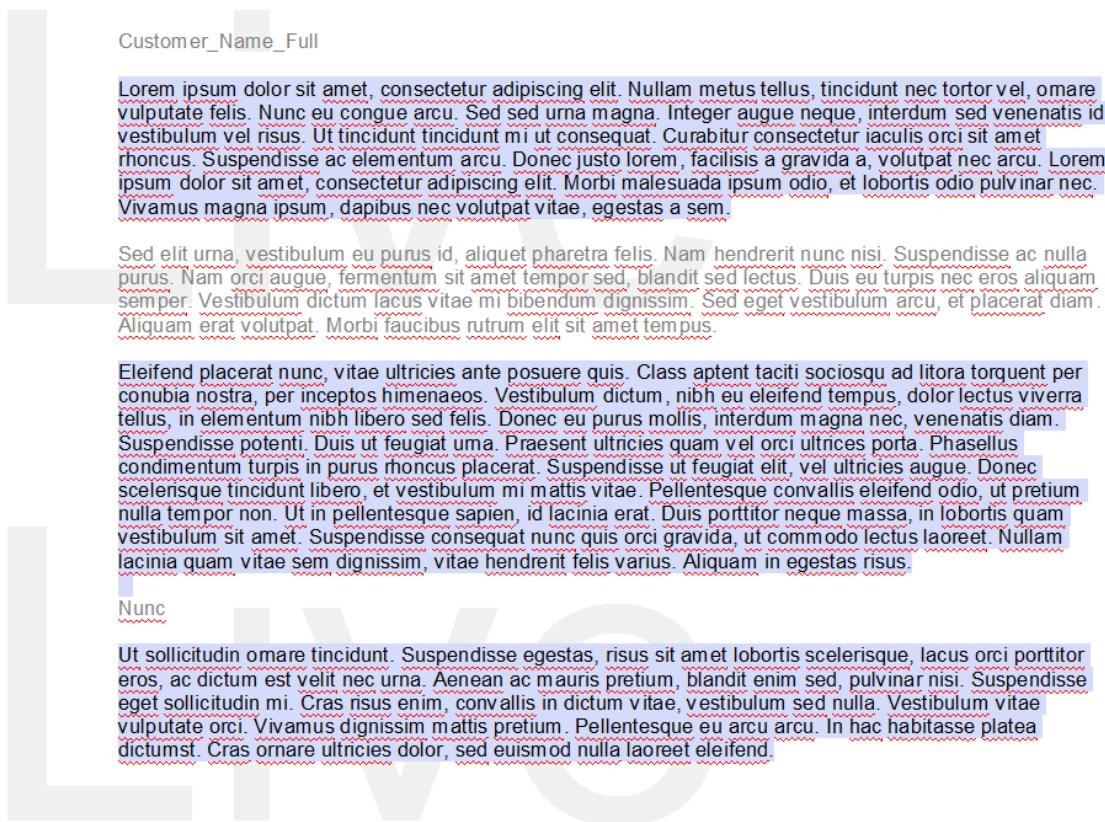
- “[Using Live Preview in Designer to Test Interactive Components](#)” on the next page
- “[Previewing Output Colors in a Live Document](#)” on page 321
- “[Locating Live Objects on a Page](#)” on page 324
- “[Using the Debugger to Test Logic in a Live Document](#)” on page 325

21.1 Using Live Preview in Designer to Test Interactive Components

Designer provides a tool called Live Preview that lets you see how the Live components of a page will appear and behave in LiveEditor. Since Live Preview is part of Designer, you can easily test and adjust the interactive components of a design in one program without having to generate a Live document to test the design in LiveEditor. For example, suppose you designed a page that contains a drop-down list. You can use Live Preview to see how the drop-down list will appear to end users. Based on the preview, you can then adjust the items that appear on the list, or you can adjust the theme to make the list be more prominent in the Live document.

For more information about themes, see “[Using Themes to Visually Guide End Users](#)” on [page 218](#).

Live Preview mode



You cannot make changes to a Live document in Live Preview mode; you can only view it and interact with its Live components. In addition, Live Preview does not enable all of the interactive

functionality in the Live document. For example, if the Live document contains a file upload area, you can see the area and how it is designated by the theme, but you cannot upload a file (since that would change the Live document). If you want to conduct more thorough testing of the interactive components of the Live document, you must generate the Live document and open it in LiveEditor. For best results, make sure you test a Live document in both Live Preview and in LiveEditor before moving it into production.

To use Live Preview in Designer to test interaction:

1. In Designer, open the page that contains the interactive component you want to test. If you want to test a theme, open any page that contains the type of component with which you want to test the theme. (For example, to test the theme conventions for required text, open a page that contains text that end users must change.)
2. On the Menu bar, select **Tools > Options**.

The **Designer Options** dialog box opens.

3. Click the **Designer** tab.
4. In the **Exstream Live Simulation Mode** area, specify the theme you want to use in Live Preview.

- a. Click .

The **Select Theme** dialog box opens.

- b. In the box, select the theme you want to use.
- c. Click **OK**.

The **Select Theme** dialog box closes and the theme you selected appears in the **Exstream Live Simulation Mode** box.

5. If you want a watermark to appear in Live Preview to help you distinguish between Live Preview mode and design mode, select the **Show Live watermark in Live mode** check box.
6. Click **OK**.

The **Designer Options** dialog box closes.

7. From the Menu bar, select **Edit > Save**.

8. On the Properties toolbar, click .

Live Preview opens.

You can interact with and test the Live document as needed in Live Preview. Click  to toggle between Live Preview mode and the design mode.

21.2 Previewing Output Colors in a Live Document

When you compare a color in on a computer monitor (such as in a Live document) against the same color in printed output, you can sometimes see variations between the two versions.

These variations occur because computer monitors use the RGB (red, green, blue) color model to display colors, and printed output uses the CMYK (cyan, magenta, yellow, black) color model.

To help compensate for the differences inherent between the two color models, Exstream lets end users temporarily change the colors of certain objects in a Live document so that the colors more closely match their CMYK counterparts. End users can sample colors from Live documents and replace them with corresponding colors from CMYK output files. This functionality is for previewing purposes only and does not permanently alter the DLF file.

This section discusses the following topics:

- [“About Previewing Output Colors in LiveEditor” below](#)
- [“Adding Colors to a Color Data File” on the next page](#)
- [“Removing Preview Colors from Your Color Data File” on page 323](#)
- [“Loading a Color Data File” on page 324](#)

21.2.1 About Previewing Output Colors in LiveEditor

LiveEditor gives end users the ability to preview output colors in a Live document. The color preview functionality lets end users see close approximations of CMYK colors during an editing session. This functionality can be particularly useful if your Live document workflow includes an approval stage before the DLF file moves to fulfillment. For example, suppose you work for an insurance agency and you design a company mailing that uses your corporate color scheme. If a manager must approve the Live document, he or she is less likely to do so if the corporate colors displayed in LiveEditor do not closely match their printed counterparts. In this scenario, the manager can temporarily replace the “incorrect” RGB corporate colors in LiveEditor with their “correct” CMYK counterparts from a CMYK output file.

LiveEditor uses an XML file (a color data file) to store information about color preview settings. However, within the RGB color model, colors can vary according to the types of monitors and graphics cards installed on end users' computers. Therefore, even identical monitors will display colors differently unless they are calibrated identically. For example, if you intend to create a single color data file to distribute with a Live document, then you must make sure not only that all end users view the Live document on equipment that exactly matches yours, but also that their monitors are calibrated identically. Otherwise, each LiveEditor user must build a separate color data file for use on his or her computer.

Note: Keep in mind that monitors use the RGB color model to display all colors. As a result, end users will likely still see slight variations between output colors on screen and corresponding output colors in print. However, even the RGB approximations that end users see on screen should still closely match the "correct" output colors.

If you do not include this functionality as part of your Live document, you can avoid confusion among LiveEditor users by removing the **Tools > Preview Output Colors** option from the LiveEditor Menu bar.

For more information about removing options from the LiveEditor Menu bar, see ["Customizing the End User Interface" on page 226](#).

Keep in mind that LiveEditor users can preview output colors only for objects that you create in Designer. For example, if you create a Live document that uses your company's color scheme, end users can change preview colors for objects such as page borders or color text boxes but not for imported images (such as corporate logos). Additionally, because this preview functionality lets end users sample colors pixel by pixel, they should use it only when working with objects made up of solid colors that do not use fill effects such as gradients.

The ability to preview output colors is available in both Designer and LiveEditor. So end users can load a color data file in LiveEditor that you created in Designer. The topics in this section describe how this functionality works in LiveEditor, but you can use the same basic procedures when working in Designer.

21.2.2 Adding Colors to a Color Data File

Before you can add colors to your color data file, you must open the Live document and the corresponding output page (such as a CMYK PDF) side-by-side. With both files open, you can easily sample colors between the two files. For example, suppose that the RGB versions of your company's colors in a Live document look different from the CMYK versions in printed output. In that scenario, you can use the colors in the Live document as source colors, and then use the corresponding colors from a CMYK PDF as your output colors. After you have added the appropriate output colors, you will then be able to preview the "correct" colors in LiveEditor.

Note: Keep in mind that monitors use the RGB color model to display all colors. As a result, you might see slight variations in color between a CMYK output file viewed on your monitor and actual CMYK printed output.

To add colors to a color data file:

1. In LiveEditor, on the Menu bar, select **Tools > Preview Output Colors**.
The **Color Preview Palette** opens.
2. Select the **Show output color preview** check box.

Any existing output colors are displayed on the design page. You can toggle between the RGB colors on your design page and the output color previews by selecting or clearing this check box.

3. In the **Output color data file** field, check the path and file name to ensure that you are using the correct color data file.

For information about loading a color data file, see [“Loading a Color Data File” on the next page](#).

4. Click the **Source color** box and then use the pointer to select the color that you want to replace in the preview.
5. Click the **Output color** box and then use the pointer to select the color that will replace the source color in the preview.

Note: You can select an output color from any program that is open and visible.

6. Click .

The source color and output color selections appear in the **Color Preview Palette**, and the output color appears on the Live document.

7. Repeat step 4 through step 6 as necessary.

21.2.3 Removing Preview Colors from Your Color Data File

As you build a color data file, if you decide that you no longer need some of the source/output color combinations that you have added, you can remove them from the output color file. For example, if the colors in your design change due to rebranding, you can remove unnecessary source/output color combinations so that the list displayed on the **Color Preview Palette** does not contain unnecessary additions.

To remove preview colors from your color data file:

1. In LiveEditor, on the Menu bar, select **Tools > Preview Output Colors**.

The **Color Preview Palette** opens.

2. Select a source color and output color combination from the list.

3. Click .

The source color and output color combination is removed from the list.

4. Repeat step 2 through step 3 as necessary.

21.2.4 Loading a Color Data File

LiveEditor stores output color preview information in an XML file called the color data file. If you intend to create a single color data file to distribute with a Live document, keep in mind that the ability to preview output colors is available in both Designer and LiveEditor. So even if you create a color data file in Designer, end users can still load it in LiveEditor.

For more information about distributing a color data file with a Live document, see “[About Previewing Output Colors in LiveEditor](#)” on page 321.

To load a color data file:

1. In LiveEditor, on the Menu bar, select **Tools > Preview Output Colors**.

The **Color Preview Palette** opens.

2. Click  to browse to a color data file.

The **Open** dialog box opens.

3. Browse to the location of the file that you want to open.
4. Click **Open**.

Note: If the file does not exist, you will be prompted to create it. On the LiveEditor dialog box that opens, click **Yes** to create the file.

The source color and output color combinations contained in the color data file appear in the **Color Preview Palette**.

21.3 Locating Live Objects on a Page

If you want to quickly locate all of the Live objects on a page, Designer provides a search feature that you can use to make all Live objects on a page appear highlighted. This feature can be helpful if you are making global adjustments to Live objects or if you want to verify that only specific objects on a page have Live properties.

To locate Live objects on a page:

1. In Designer, open the page you want to search.
2. On the Menu bar, select **Edit > Search > Interactive areas**.

All of the Live objects on the page are highlighted.

21.4 Using the Debugger to Test Logic in a Live Document

Live documents often contain complex logic that supports intelligence or automation within the document. Exstream Live provides a tool called the Debugger that you can use to analyze and test the logic and events in a Live document. The Debugger is located in LiveEditor; therefore, as you test a Live document in LiveEditor, you can step through the process an end user might use to edit a document and verify the logic at each point in the editing process.

Tip: Because the Debugger is a tool most often used by Live document designers and programmers, you might want to use the view object to prevent it from appearing in the LiveEditor for end users.

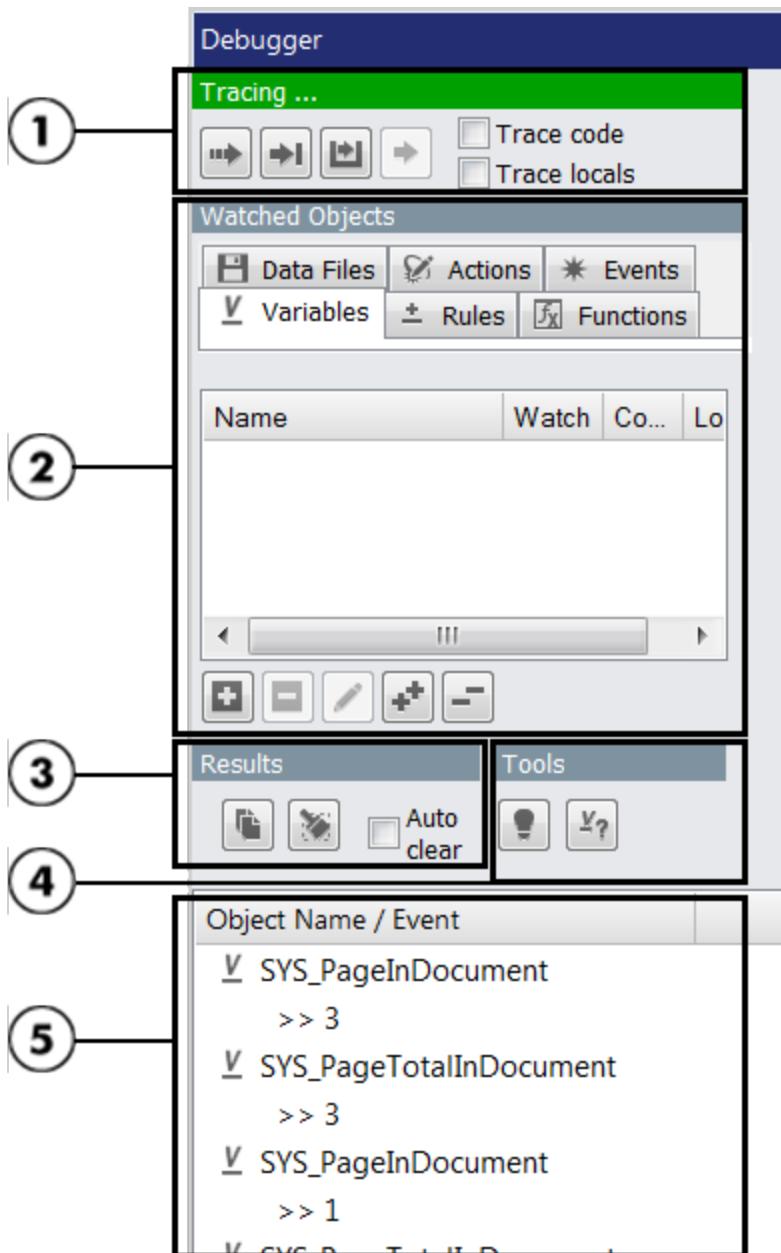
This section discusses the following topics:

- “[The Debugger at a Glance](#)” below
- “[Controlling How Results Appear](#)” on page 327
- “[Controlling the Level of Detail Recorded for Rules, Formulas, and Functions](#)” on page 331
- “[Testing Specific Objects in a Live Document](#)” on page 332
- “[Viewing the Current Value of Variables](#)” on page 343
- “[Viewing the Design Properties of Objects](#)” on page 345
- “[Sharing Debugger Results](#)” on page 346

21.4.1 The Debugger at a Glance

You can show or hide the Debugger by selecting **View > Debugger** from the Menu bar in LiveEditor. Because the Debugger is a panel, you can move and dock it to make it easier to use.

The following illustration calls out the main areas of the Debugger and describes how you use them.

Debugger at a glance

1	Tracing area—Control how information appears in the results list and the level of detail that appears there.
2	Watched Objects area—Specify the objects you want to test.

(3)	Results area—Clear the current results in the results list or copy them so you can use them elsewhere.
(4)	Tools area—View the current value of variables or the design properties of objects.
(5)	Results list—View the information reported by the Debugger.

21.4.2 Controlling How Results Appear

You have several options for controlling the way results appear in the Debugger. Different result options are better suited for different types of testing. For example, if you are stepping through the workflow you expect an end user to use in a Live document, you might want the Debugger to simply record the results of each action you take in a long list, which you can later analyze. On the other hand, suppose you are testing a complex Trigger function. In this case, you might want each line of code in the function to appear one at a time, so you can analyze what is happening during each step of the function execution. This type of reporting is called "single-stepping." Some result options are affected by watches placed on objects; for example, the "Run until an object watch is reached" option includes results only until a watched object is reached.

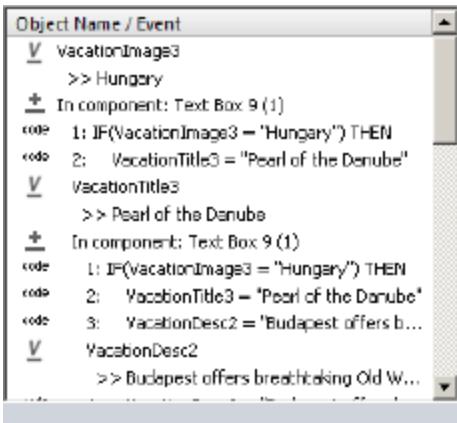
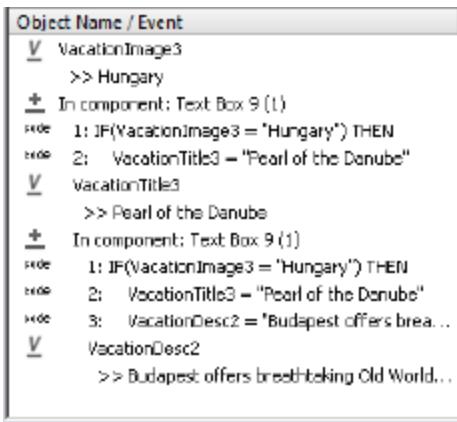
For information about object watches, see ["Testing Specific Objects in a Live Document" on page 332](#).

You can also control how much detail appears in the results for rules, formulas, and functions. This information is discussed in a separate task.

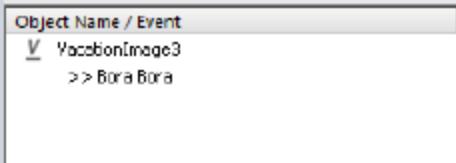
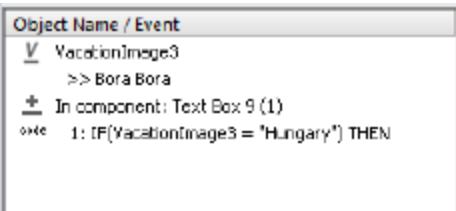
For information about controlling the level of detail that appears in the results list, see ["Controlling the Level of Detail Recorded for Rules, Formulas, and Functions" on page 331](#).

The following table lists the ways results can appear in the results list:

Options for displaying Debugger results

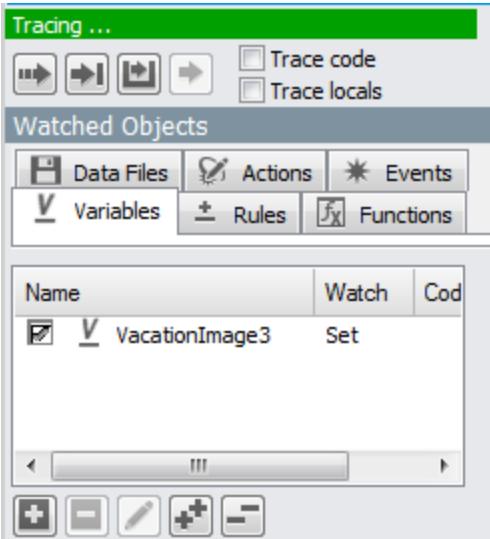
Option	Description	Example
Run without stopping (also called tracing)	This method records all logic information at one time as you interact with the Live document. It stops recording information only when you stop making selections. This method is useful if you are walking through the actions an end user will take and you want to quickly see a complete list of all the logic that will be executed during editing.	<p>In the following example, a watch is placed on the 'VacationDesc2' variable; however, you can see that the information continues to be recorded even after that variable is set, and you can use the scrollbar to browse through all of the recorded information.</p> <p>Example of "run without stopping" results</p>  <pre> Object Name / Event V VacationImage3 >> Hungary + In component: Text Box 9 (1) code 1: IF(VacationImage3 = "Hungary") THEN code 2: VacationTitle3 = "Pearl of the Danube" V VacationTitle3 >> Pearl of the Danube + In component: Text Box 9 (1) code 1: IF(VacationImage3 = "Hungary") THEN code 2: VacationTitle3 = "Pearl of the Danube" code 3: VacationDesc2 = "Budapest offers b... V VacationDesc2 >> Budapest offers breathtaking Old W... </pre>
Run until an object watch is reached	This method records all the logic information at one time as you interact with the Live document. It stops recording information when a watched object or event is encountered. This method is useful if the object you are testing is affected by other events in the Live document. For example, if a reference file read is triggered based on an end user's selection from a list, you might want the Debugger to stop recording information when the read is triggered so you can see what events caused the read. For information about setting up a watch, see "Testing Specific Objects in a Live Document" on page 332.	<p>In the following example, a watch is placed on the 'VacationDesc2' variable. No information is recorded in the results list after that variable is set, even if other events are set up to occur automatically.</p> <p>Example of "run until an object watch is reached" results</p>  <pre> Object Name / Event V VacationImage3 >> Hungary + In component: Text Box 9 (1) code 1: IF(VacationImage3 = "Hungary") THEN code 2: VacationTitle3 = "Pearl of the Danube" V VacationTitle3 >> Pearl of the Danube + In component: Text Box 9 (1) code 1: IF(VacationImage3 = "Hungary") THEN code 2: VacationTitle3 = "Pearl of the Danube" code 3: VacationDesc2 = "Budapest offers b... V VacationDesc2 >> Budapest offers breathtaking Old World... </pre>

Options for displaying Debugger results, continued

Option	Description	Example
Single-step	This method records logic one line at a time so you can easily analyze what occurs when complex logic is being performed. Single-stepping is particularly useful when you are troubleshooting a complex function or event.	<p>When you start the single-step process, the first line of logic appears in the Debugger, as in the following example:</p> <p>Example of first step in "single-step" result</p>  <p>When you advance the Debugger, another line of logic is added to the results list:</p> <p>Example of second step in "single-step" result</p> 

To control how results appear, select one of the following options in the tracing area:

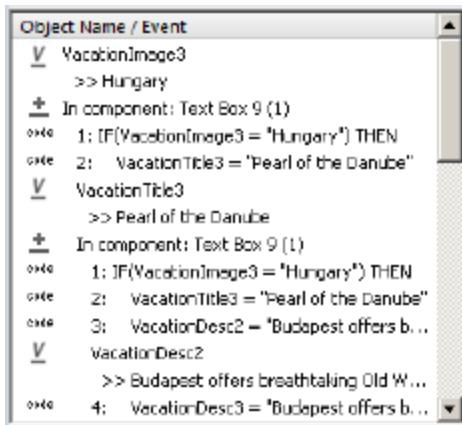
To	Do this
Run without stopping	<p>Click .</p> <p>As you interact with the Live document, the results of all of your actions appear in the results list. The trace does not stop until you stop making selections.</p>

To	Do this
Run until a watch is reached	<p>Click .</p> <p>As you interact with the Live document, the results of all of your actions appear in the results list until the watch is reached. Then, the watched object appears in red in the Debugger to indicate it has been reached, and the Debugger stops recording information.</p> <p>Example of watched object reached</p>  <p>You cannot interact with the Live document until you either clear the results by clicking  or you indicate you want the trace to continue by clicking .</p> <p>For information about setting up a watch, see “Testing Specific Objects in a Live Document” on page 332.</p>
Single-step	<p>Click .</p> <p>As you interact with the Live document, the results of your actions appear in the results list one line at a time. To see the next line of code in the results list, click  or .</p>

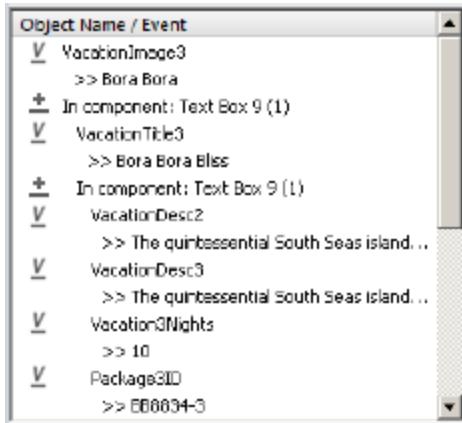
21.4.3 Controlling the Level of Detail Recorded for Rules, Formulas, and Functions

The Debugger lets you control whether or not details about rules, formulas, and functions appear in the results list. For example, if you allow the details to be included, the logic that makes up a text rule appears in the results list when that rule is executed. This level of detail is called "code trace." However, if you disable the detail, the results list only includes the name of the rule that was executed.

Example of results with code trace



Example of results without code trace



You can also control whether local variables (variables used within logic and not part of the Data Dictionary) are listed in the results list.

To control the level of detail recorded for rules, formulas, and functions, select one of the following options in the tracing area:

To	Do this
Include the code information for rules, formulas, and functions	Select the Trace code check box. The code information appears in the results list with the identifier code.
Include information about local variables	Select the Trace locals check box.

If you include the code information for rules, formulas or functions, you can double-click the object in the results list to open the **Code** dialog box. The **Code** dialog box displays the logic executed during that step.

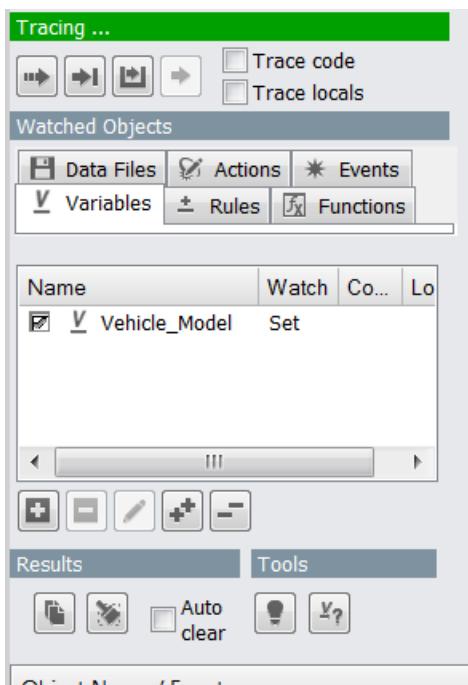
21.4.4 Testing Specific Objects in a Live Document

If you want to see how specific objects in a Live document behave during editing, you can use the Debugger to track and report information about those specified objects. You can test the following types of objects in a Live document:

- Data files
- Actions
- System events
- Variables
- Rules
- Functions

You test specific objects by instructing the Debugger to watch them (sometimes called "placing a watch on an object"). When an object is "watched," the Debugger collects information about the object and displays it in the results list. For example, in the following illustration, the variable named 'VacationImage3' is selected as the object for the Debugger to watch. When you set the Debugger to run until a watch is reached, it records information in the results list until it reaches a watched object or event and then stops.

Watched variable in the Debugger



The Debugger also lets you watch all of a specific type of object. For example, you can quickly set up a watch on all the actions within a Live document.

For information about using the tracing options to control how information appears in the results list, see “[Controlling How Results Appear](#)” on page 327.

To test specific objects in a Live document, complete the following tasks as needed:

- “[Watching a Data File](#)” on the next page
- “[Watching an Action](#)” on page 335
- “[Watching an Event](#)” on page 336
- “[Watching a Variable](#)” on page 337
- “[Watching a Rule](#)” on page 339
- “[Watching a Function](#)” on page 342

Watching a Data File

1. On the Debugger, in the **Watched Objects** area, select the **Data File** tab.
2. Depending on whether you want to watch a specific data file or all data files, complete one of the following tasks at the bottom of the **Watched Objects** area:

To	Do this
Watch a specific data file	<p>a. Click  .</p> <p>The Data File - Add watch dialog box opens.</p> <p>b. In the Object to watch box, click  .</p> <p>The Select Data File dialog box opens.</p> <p>c. Select the data file you want to test and click OK.</p> <p>The Select Data File dialog box closes and the data file you selected appears in the Object to watch box.</p> <p>d. Click OK.</p> <p>The Data File - Add Watch dialog box closes and the data file you selected appears in the Watched Objects box.</p>
Watch all data files in the Live document	<p>a. Click  .</p> <p>The SBCS Data Files - Watch all dialog box opens.</p> <p>b. Click OK.</p> <p>The SBCS Data Files - Watch all dialog box closes and a list of all the data files in the Live document appears in the Watched Objects box.</p>

If you want to turn the watch off temporarily (but retain the watch settings so you can use them again later), clear the **Enabled** check box on the **Data File - Add Watch** dialog box or on the **SBCS Data Files - Watch all** dialog box.

3. From the Menu bar, select **File > Save**.

Watching an Action

1. On the Debugger, in the **Watched Objects** area, select the **Actions** tab.
2. Depending on whether you want to watch a specific action or all actions, complete one of the following tasks at the bottom of the **Watched Objects** area:

To	Do this
Watch a specific action	<ol style="list-style-type: none">a. Click  . The Action - Add watch dialog box opensb. In the Object to watch box, click  . The Select Function dialog box opens.c. Select the action you want to test and click OK. The Select function dialog box closes and the action you selected appears in the Object to watch box.d. Click OK. The Action - Add Watch dialog box closes and the action you selected appears in the Watched Objects box.
Watch all actions in the Live document	<ol style="list-style-type: none">a. Click  . The Actions - Watch all dialog box opens.b. Click OK. The Actions - Watch all dialog box closes and a list of all the actions in the Live document appears in the Watched Objects box.

If you want to turn the watch off temporarily (but retain the watch settings so you can use them again later), clear the **Enabled** check box on the **Action - Add Watch** dialog box or on the **Actions - Watch all** dialog box.

3. From the Menu bar, select **File > Save**.

Watching an Event

1. On the Debugger, in the **Watched Objects** area, select the **Events** tab.
2. Depending on whether you want to watch a specific event or all events, complete one of the following tasks at the bottom of the **Watched Objects** area:

To	Do this
Watch a specific event	<p>a. Click .</p> <p>The System Event- Add watch dialog box opens.</p> <p>b. From the Object to watch drop-down list, select the event you want to watch:</p> <ul style="list-style-type: none"> • When document closed—Any time an end user closes the Live document • When document opened—Any time an end user opens the Live document • When document saved—Any time an end user saves the Live document • Initial document close—The first time an end user closes the Live document • Initial document open—When an end user opens the Live document for the first time • Initial document save—When an end user saves the Live document for the first time • Customer start—At the beginning of new customer data • Customer end—At the end of new customer data • When data read—When an end user reads a data file • When data written—When an end user writes to a data file • On ODBC record update—When an end user updates an ODBC database record • On ODBC record delete—When an end user deletes an ODBC database record • When stored—When an end user stores the Live document • When submitted—When an end user submits the Live document • When printed—When an end user prints the Live document <p>c. Click OK.</p> <p>The System Event - Add Watch dialog box closes and the event timing you selected appears in the Watched Objects dialog box.</p>
Watch all events in the Live document	<p>a. Click .</p> <p>The System Events - Watch all dialog box opens.</p> <p>b. Click OK.</p> <p>The System Events - Watch all dialog box closes and a list of all the events in the Live document appears in the Watched Objects box.</p>

If you want to turn the watch off temporarily (but retain the watch settings so you can use them again later), clear the **Enabled** check box on the **System Events' Add Watch** dialog box or on the **System Events - Watch all** dialog box.

3. From the Menu bar, select **File > Save**.

Watching a Variable

1. On the Debugger, in the **Watched Objects** area, select the **Variables** tab.
2. Depending on whether you want to watch a specific variable or all variables, complete one of the following tasks at the bottom of the **Watched Objects** area:

To	Do this
Watch a specific variable	<p>a. Click .</p> <p>The Variable - Add watch dialog box opens.</p> <p>b. In the Object to watch box, click .</p> <p>The Select Variable dialog box opens and displays a list of all the variables in the Live document.</p> <p>c. Select the variable you want to watch and click OK.</p> <p>The Select Variable dialog box closes and the variable you selected appears in the Object to watch box.</p> <p>d. If you want the selected variable to be watched only under certain conditions, select one of the following conditions from the Watch type drop-down list:</p> <ul style="list-style-type: none"> • None—The variable is never watched. • Set—The variable is watched when its initial value is set. • Change—The variable is watched each time its value changes. • Match—The variable is watched when its value matches a specified value. <p>e. If you selected Match from the Watch type drop-down list, enter the variable value in the Match value box that the variable must have in order to be watched.</p> <p>f. If you want to override the default code trace options for this variable, select one of the following options from the Code trace drop-down list:</p> <ul style="list-style-type: none"> • None—No code is traced. • Trace—All code is traced. • Single-step—Code is traced in single steps. <p>g. If you want to override the default local variable watch options for this variable, select one of the following options from the Local variable watch drop-down list:</p> <ul style="list-style-type: none"> • None—Local variables are not watched. • Trace—The watch is performed when values are assigned to local variables. • Watch—In addition to tracing the local variable value, the Debugger stops after each local variable trace. <p>h. Click OK.</p> <p>The Variable - Add watch dialog box closes and the variable you selected appears in the Watched Objects box.</p>

To	Do this
Watch all variables in the Live document	<p>a. Click . The Variables - Watch all dialog box opens.</p> <p>b. If you want the variables to be watched only under certain conditions, select one of the following conditions from the Watch type drop-down list:</p> <ul style="list-style-type: none">• None—The variables are never watched.• Set—The variables are watched when its initial value is set.• Change—The variables are watched each time its value changes.• Match—The variables are watched when its value matches a specified value. <p>c. If you selected Match from the Watch type drop-down list, enter the variable value in the Match value box that the variables must have in order to be watched.</p> <p>d. Click OK. The Variables - Watch all dialog box closes and a list of all the variables in the Live document appears in the Watched Objects box.</p>

3. From the Menu bar, select **File > Save**.

Watching a Rule

1. On the Debugger, in the **Watched Objects** area, select the **Rules** tab.
2. Depending on whether you want to watch a specific rule or all rules, complete one of the following tasks at the bottom of the **Watched Objects** area:

To	Do this
Watch a specific rule	<p>a. Click  .</p> <p>The Rule - Add watch dialog box opens.</p> <p>b. In the Object to watch box, click  .</p> <p>The Select Rule dialog box opens and displays a list of all the rules in the Live document.</p> <p>c. Select the rule you want to watch and click OK.</p> <p>The Select Rule dialog box closes and the rule you selected appears in the Object to watch box.</p> <p>d. If you want the selected rule to be watched only under certain conditions, select one of the following conditions from the Watch type drop-down list:</p> <ul style="list-style-type: none"> • None—The rule is never watched. • Execute—The rule is watched when it is executed. • Change—The rule is watched when the value changes. • Match—The rule is watched when it matches a specified value. <p>e. If you selected Match from the Watch type drop-down list, enter the rule value in the Match value box that the rule must have in order to be watched.</p> <p>f. If you want to override the default code trace options for this rule, select one of the following options from the Code trace drop-down list:</p> <ul style="list-style-type: none"> • None—No code is traced. • Trace—All code is traced. • Single-step—Code is traced in single steps. <p>g. If you want to override the default local variable watch options for this rule, select one of the following options from the Local variable watch drop-down list:</p> <ul style="list-style-type: none"> • None—Local variables are not watched. • Trace—The watch is performed when values are assigned to local variables. • Watch—In addition to tracing the local variable value, the Debugger stops after each local variable trace. <p>h. Click OK.</p> <p>The Rule - Add watch dialog box closes and the variable you selected appears in the Watched Objects box.</p>

To	Do this
Watch all rules in the Live document	<p>a. Click  .</p> <p>The Rules - Watch all dialog box opens.</p> <p>b. If you want the rules to be watched only under certain conditions, select one of the following conditions from the Watch type drop-down list:</p> <ul style="list-style-type: none">• None—The rules are always watched.• Execute—The rules are watched when they are executed.• Change—The rules are watched when their value changes.• Match—The rules are watched when they match a specified value. <p>c. If you selected Match from the Watch type drop-down list, enter the rule value in the Match value box that the rules must have in order to be watched.</p> <p>d. If you want to override the default code trace options for all of the rules, select one of the following options from the Code trace drop-down list:</p> <ul style="list-style-type: none">• None—No code is traced.• Trace—All code is traced.• Single-step—Code is traced in single steps. <p>e. If you want to override the default local variable watch options for all of the rules, select one of the following options from the Local variable watch drop-down list:</p> <ul style="list-style-type: none">• None—Local variables are not watched.• Trace—The watch is performed when values are assigned to local variables.• Watch—In addition to tracing the local variable value, the Debugger stops after each local variable trace. <p>f. Click OK.</p> <p>The Rules - Watch all dialog box closes and a list of all the rules in the Live document appears in the Watched Objects box.</p>

3. From the Menu bar, select **File > Save**.

Watching a Function

1. On the Debugger, in the **Watched Objects** area, select the **Functions** tab.
2. Depending on whether you want to watch a specific function or all functions, complete one of the following tasks at the bottom of the **Watched Objects** area:

To	Do this
Watch a specific function	<p>a. Click .</p> <p>The Function - Add watch dialog box opens.</p> <p>b. In the Object to watch box, click .</p> <p>The Select Function dialog box opens.</p> <p>c. Select the function you want to test and click OK.</p> <p>The Select Function dialog box closes and the function you selected appears in the Object to watch box.</p> <p>d. If you want to override the default code trace options for this function, select one of the following options from the Code trace drop-down list:</p> <ul style="list-style-type: none">• None—No code is traced.• Trace—All code is traced.• Single-step—Code is traced in single steps. <p>e. If you want to override the default local variable watch options for this function, select one of the following options from the Local variable watch drop-down list:</p> <ul style="list-style-type: none">• None—Local variables are not watched.• Trace—The watch is performed when values are assigned to local variables.• Watch—In addition to tracing the local variable value, the Debugger stops after each local variable trace. <p>f. Click OK.</p> <p>The Function - Add watch dialog box closes and the function you selected appears in the Watched Objects box.</p>

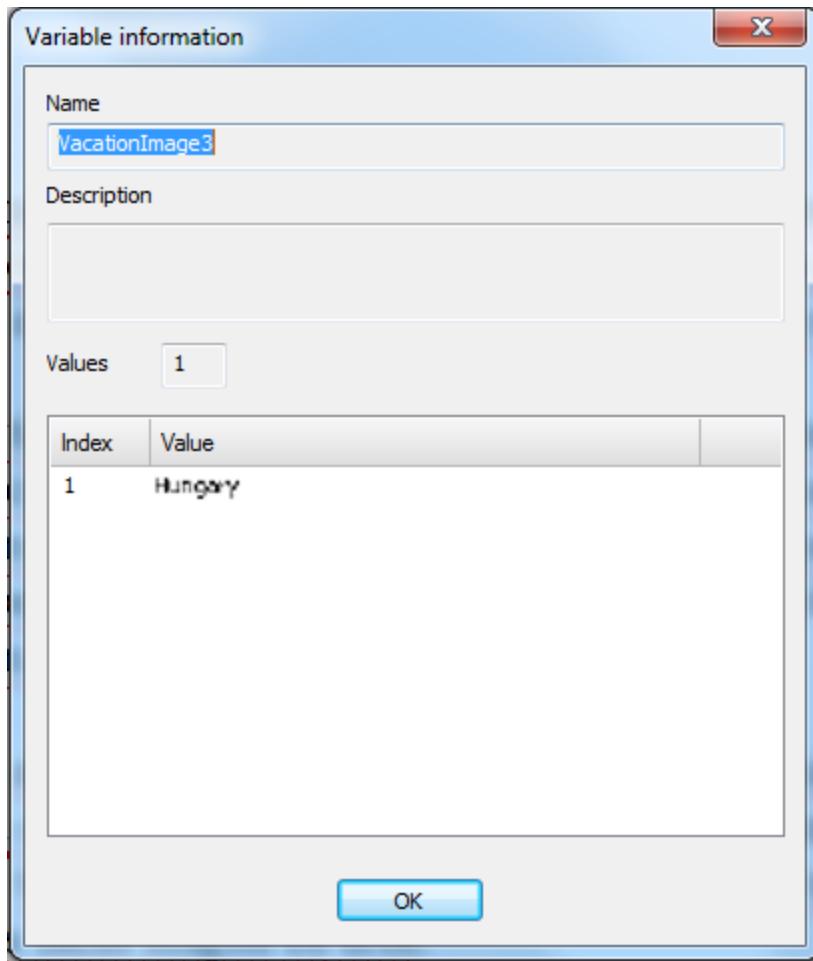
To	Do this
Watch all functions in the Live document	<p>a. Click  .</p> <p>The Functions - Watch all dialog box opens.</p> <p>b. If you want to override the default code trace options for all of the functions, select one of the following options from the Code trace drop-down list:</p> <ul style="list-style-type: none">• None—No code is traced.• Trace—All code is traced.• Single-step—Code is traced in single steps. <p>c. If you want to override the default local variable watch options for all of the functions, select one of the following options from the Local variable watch drop-down list:</p> <ul style="list-style-type: none">• None—Local variables are not watched.• Trace—The watch is performed when values are assigned to local variables.• Watch—In addition to tracing the local variable value, the Debugger stops after each local variable trace. <p>d. Click OK.</p> <p>The Functions - Watch all dialog box closes and a list of all the functions in the Live document appears in the Watched Objects box.</p>

3. From the Menu bar, select **File > Save**.

21.4.5 Viewing the Current Value of Variables

The Debugger lets you quickly see the current value of variables in a customer. This feature can be helpful if you are troubleshooting an event dependent on a variable or setting up a watch that requires a specific variable value.

Variable values for the current customer



To view the current value of variables:

1. On the Debugger, in the **Tools** area, click .

The **Select Variable** dialog box opens and displays a list of all the variable values in the current customer. To filter variables based on their type, click .

2. When you are finished, click **OK**.

The **Select Variable** dialog box closes. The **Variable information** dialog box opens and displays information about the variable you selected.

3. Click **OK**.

The **Variable information** dialog box closes.

4. From the Menu bar, select **File > Save**.

21.4.6 Viewing the Design Properties of Objects

The Debugger provides a feature called Inspection Mode that allows you to right-click on any object with Live properties and view all of the design property settings of the object. You might find this feature helpful if you are setting up a watch on a rule, formula, or function and you want to see the variables associated with it. For example, suppose you want to set up a watch on the variable associated with a check box. After enabling Inspection Mode, you can right-click the check box and view all of its design properties, including the variable assigned to it. When you turn on Inspection Mode, all of the interactive areas and objects in the Live document are designated with colored underlines so you can see, at a glance, which objects are Live.

Inspection Mode off

Fund Category	Fund Name
-Select-	-Select Fund-

Inspection Mode on

Fund Category	Fund Name
-Select-	-Select Fund-

To view the design properties of objects:

1. On the Debugger, in the **Tools** area, click .

Inspection Mode is enabled. Interactive areas and Live objects are underlined with colored lines.

2. Right-click the object for which you want to view properties.

A shortcut menu opens. Most objects have only one option. Other objects, such as variables, have more than one option for the types of properties you can view.

3. Select the appropriate option from the shortcut menu.

The object's properties open.

4. From the Menu bar, select **File > Save**.

You can browse the object's properties, but you cannot edit them. To make changes to the properties, you must edit the Live document in Designer.

21.4.7 Sharing Debugger Results

If you want to save or share the results of a Debugger test with someone else, the Debugger lets you easily copy a text version of the results. You can paste the results into a text file, email, or other document and save or share them.

When you copy the results of the Debugger, icons that appear in the results list are replaced with the following text codes:

Debugger text codes

Object	Text code
Action	A> and A<
Data file	D> and D<
Event	E> and E<
Function	F> and F<
Rule	R> and R<
Formula variable	V> and V<
Variable	V:
Code (logic contained within a formula variable, rule, or function)	C:
Local variable	V:
Debugger message, informational	I
Debugger message, question	Q
Debugger message, warning	W
Debugger message, error	E

To share Debugger results:

1. On the Debugger, in the **Tools** area, click .

The information in the results list are copied to the clipboard.

2. Paste the Debugger information into the program of your choice.

Debugger results in text form

```
F> triggerCustomer
V:     clicked
      >> TRUE
F<
R> On component: Text 44
      >> 1
V: ChangeAddress
      >> TRUE
V: Purchase_Catagory_Selection1
      >> 2
V: Purchase_catagory
      [1] >> Growth-and-Income funds
R> In component: Text 31 (16)
V: Funds_Display1
      [1] >> Exstream Mutual Fund
R> In component: Text 31 (16)
V: Funds_Display1
      [2] >> Capital World Growth and Income Fund
V: Funds_Display1
      [3] >> Fundamental Investors
V: Funds_Display1
      [4] >> The Investment Company of America
V: Funds_Display1
      [5] >> Washington Mutual Investors Fund
V: Purchase_Catagory_Selection1PREV
      >> 2
R<     >> 1
      >> 1
```

Chapter 22: Generating Live Documents

During the output and production phase, you generate output and the Live document is made available for use. The properties you define for the output object depend on the decisions that you made during the planning and analysis, data, and design phases. For example, suppose you determined during the planning and analysis phase that you wanted to use visual indicators to inform users of the actions they can take. Based on that decision, you subsequently created the interactive features during the design phase. During the output and production phase, you will set up the properties for your Live document to use the appropriate interactive features.

After you have generated a Live document, the interaction cycle can begin. The properties you define for the Live document determine what types of interaction can take place.

This chapter discusses the following topics:

- “[Including the Appropriate Resources in a Package File](#)” below
- “[Special Timing Considerations for Live Documents](#)” on page 350
- “[Setting Up the Objects Necessary for Live Output](#)” on page 354

22.1 Including the Appropriate Resources in a Package File

In Exstream Live, applications serve the same purpose as they do in the Exstream Design and Production environment: they contain the documents, pages, and other content produced by the engine. Packaging an application is the first step to producing output. Creating a package file for your application(s) lets you combine all of the objects or elements into one format that is optimized for the engine.

This section discusses the following topics:

- “[Determining Which Resources to Include in a Package File](#)” on the next page
- “[Specifying Which Resources to Include in a Package File](#)” on the next page

22.1.1 Determining Which Resources to Include in a Package File

As you begin to plan the objects that make up your Live application, it is important to consider the objects that will be required during the editing session. For example, are there some documents that do not need to be visible in the original Live document but need to appear based on the end user's selections? By planning carefully the objects that must be available in LiveEditor or during fulfillment, you can help ensure that your Live document is streamlined and does not include unnecessary content, but still meets your end users' needs. You can control the inclusion of documents and sections or paragraphs to achieve this end.

To help ensure you have the correct documents or paragraphs included in your application, consider the following questions:

- **Have I included all the objects necessary for the editing?**—Keep in mind that some objects might not be required during the editing session initially. However, if these objects are affected by changes the end user makes or contain data that is necessary for logic processing, you must include these objects in the Live document (even if they are hidden). For example, suppose your property and casualty insurance company provides auto, boat, and homeowners insurance. If you create a Live document containing policy information that a CSR will complete when a customer calls to purchase insurance, the Live document must include documents for all types of policies (because you do not know which documents will be necessary for each customer).
- **Have I included unnecessary objects?**—Keep in mind that some objects might not need to be available during the editing session. For example, suppose your property and casualty insurance company provides auto, boat, and homeowners insurance, and you are creating a Live document that contains a letter and updated policy information for all existing customers. Since you know which policy documents each customer needs, you can make sure each customer's document includes only the required documents. When the insurance agent (end user) adds personalized text to the letter, he or she will not have to make any further selections. In addition, the DLF file size will not be larger than necessary, since unnecessary content was excluded from it.

22.1.2 Specifying Which Resources to Include in a Package File

In Exstream Live, you can set application properties that control all of the output properties that are available to traditional applications. Additionally, you also control the inclusion of additional font and image resources necessary for final production. When you include additional resources in the package file, Exstream places all images and font information from target applications in the package file when you package the application.

To specify which resources to include in a package file:

1. In Design Manager, from the Library, drag the application you want to package to the Property Panel.
2. Click the **Interactive** tab.
3. From the **Resources to package for Live document postprocessing** drop-down list, select the appropriate setting:
 - **None**—No additional resources are packaged.
 - **All applications**—Exstream includes resources from every application in your design database in the package file when you package the application.
 - **Applications in list**—Exstream includes resources from only the applications you specify. The **Interactive applications to be processed** box becomes available if you select this option. Drag applications to the box from the Library or click  to select them.
 - **Applications in folders in list**—Exstream includes resources from only the applications in the folders you specify. The **Interactive applications to be processed** box becomes available if you select this option. Drag folders to the box from the Library or click  to select them.
4. From the Menu bar, select **File > Save**.

22.2 Special Timing Considerations for Live Documents

In Exstream Live applications, variables might have different substitution times than they can have in traditional Exstream Design and Production applications. For example, data might change due to selections made by the end user in the Live document in LiveEditor, and then again during the final engine run when a DLF file is imported into another design to create the final customer document. In this case, variables must be updated several times throughout the Live document life cycle, rather than being updated only once (when document is originally created).

Additionally, it is important to include in the package file only those objects that will be required during the editing session when creating Live documents. By including only necessary objects in your Live document, you can improve the engine processing time, decrease the file size of the Live document, and improve the performance of the Live document in LiveEditor.

This section discusses the following topics:

- “[Setting the Default Variable Substitution and Rule Execution Time](#)” below
- “[Using Rules to Specify Execution Timing and Control Object Inclusion](#)” on the next page
- “[Overriding the Default Variable Substitution and Rule Execution Time for Specific Objects](#)” on page 354

For more information about rule timing, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

22.2.1 Setting the Default Variable Substitution and Rule Execution Time

You can define a default variable substitution and rule execution time using the Live setting object. The time specified on the setting object serves as the substitution time for all variables that do not have different substitution times specified. You typically set the default variable substitution time so that all variables are updated only once: during the initial engine run. By setting the default substitution time to occur only once, you can improve performance time in LiveEditor (since all the variables in a Live document do not need to be updated during the LiveEditor session). You can override the default substitution times for individual variables as needed and you can specify the default time when rules are run.

To set the default variable substitution and rule execution time:

1. In Design Manager, from the Library, drag the Live setting object to the Property Panel.
2. Click the **Editor Framework** tab.
3. From the **Default variable substitution & rule execution time** drop-down list, select one of the following options to specify when variables and rules are executed:
 - **Initial engine run only**—Variables are substituted and rules are run only during the first engine run when the Live document is produced.
 - **Initial Engine and Interactive Editor**—Variables are substituted and rules are run during the first engine run and during editing in LiveEditor. End users can preview their data in LiveEditor. This option is the default.
 - **Initial Engine, Interactive Editor, and Final Engine**—Variables are substituted and rules are run during the first and last engine run and during editing in LiveEditor.
4. From the Menu bar, select **File > Save**.

Defining the Substitution Time for a Specific Variable

If you do not want to change the default substitution time for all variables in a Live document, you can change substitution timing for individual variables instead.

To define the substitution time for a specific variable:

1. In Designer, right-click the variable for which you must set the substitution time, and select **Variable properties**.

The **Variable Properties** dialog box opens.

2. On the **Variable Use** tab, from the **When to substitute** drop-down list, select one of the following substitution times:

- **Use Settings default**—Data is read at the time specified on the **Default variable substitution & rule execution time** drop-down list on the setting object properties. This setting is the default timing.
 - **Initial engine run only**—Data is read only during the first engine run when the Live document is produced. If you select this option, end users cannot upload and use their own distribution lists.
 - **Initial Engine and Interactive Editor**—Data is read during the first engine run and when end users upload distribution lists in LiveEditor.
 - **Initial Engine, Interactive Editor, and Final Engine**—Data is read during the first and last engine run and when end users upload distribution lists in LiveEditor. Users can preview their data in the LiveEditor. You can also use that data during the final production run.
3. Click **OK**.

The **Variable Properties** dialog box closes.

4. From the Menu bar, select **Edit > Save**.

22.2.2 Using Rules to Specify Execution Timing and Control Object Inclusion

If you use rules to control object inclusion, you can use execution timing as a method to ensure that the Live document does not contain unnecessary objects when it is created.

Note: As you apply rules to a Live document, keep in mind that adding text rules to editable text can produce unexpected results in LiveEditor. For example, if an end user deletes text that is included according to a text rule, even though the deleted text will not be visible in LiveEditor, it remains part of the DLF file. In this scenario, if your text rule dictates that the editable text be included, the text will appear in the Live document even if the end user deletes it in LiveEditor. As a result, you should use text rules only for text that cannot be edited in LiveEditor.

To set the execution timing on an object:

1. In Design Manager, from the Library, drag the object on which you want to set execution timing to the Property Panel.
2. Click the **Targeting** tab.
3. Click the **Rule** box.
The **Rule** dialog box opens.
4. Click  to toggle to the code view of the **Rule** dialog box.
5. From the **Execute time** drop-down list, select the time at which you want the engine to read the rule:

Option	Description	Use when
Use Edit Settings default	This is the default. Rules are run according to the timing specified on the Default variable substitution & rule execution time drop-down list on the setting object.	The default timing option you define on the setting object is the appropriate timing option for this object.
Initial engine run only	Rules are run only during the first engine run when the Live document is created.	<p>The object is not required in LiveEditor. Use this option whenever possible because it will prevent necessary objects from being included in the Live document.</p> <p>Do not use this option if you want to allow users to upload and use their own distribution lists.</p>
Initial Engine and Interactive Editor	Rules are run during the first engine run and when users upload distribution lists in LiveEditor.	The object will be added or selected when the end user edits the document.
Initial Engine, Interactive Editor, and Final Engine	Rules are run during the first and last engine run, and when users upload mailing lists in LiveEditor. Users can preview their data in LiveEditor. You can also use that data during the final production run.	The object will be added or selected in LiveEditor and/or added to the final output (fulfillment).

6. Click **OK**.

The execution timing is applied to the selected object.

7. From the Menu bar, select **File > Save**.

For more information about creating rules and defining their properties, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

22.2.3 Overriding the Default Variable Substitution and Rule Execution Time for Specific Objects

In some cases, your application might contain many documents with inclusion rules that are executed at the initial engine run and in LiveEditor, but only some of the documents must be included in a particular Live document. In such a case, you can use an array variable to restrict the objects that are eligible for inclusion in the application. To use this method, complete one of the following procedures, depending on whether you are controlling the inclusion of documents or sections/paragraphs.

To	Do this
Use a variable to control the inclusion of documents	<ol style="list-style-type: none">1. Create an array variable that provides a list of all the documents you want to include in the Live document. You can provide a static list or use a formula to determine the list based on customer data.2. On the Documents tab of the application properties, select this variable in the List of documents to send to customer (used for variable-driven assembly applications) box.3. On the Targeting tab of the document properties, from the Method drop-down list in the Document inclusion method area, select one of the Document-assembly variable [checks or ignores rule] options.4. If you want to use a rule to further control the object inclusion, provide the rule in the Rule box.
Use a variable to control the inclusion of sections or paragraphs	<ol style="list-style-type: none">1. Create an array variable that provides a list of all the sections or paragraphs you want to include in the Live document. You can either provide a static list or use a formula to determine the list based on customer data.2. On the Targeting tab of the document properties, from the Method drop-down list in the Paragraph inclusion method area, select one of the Variable selects [object] options.3. If you want to use a rule to further control the object inclusion, provide the rule in the Targeting Rule box on the Targeting tab of the section or paragraph properties.

For information about using the rule execution timing options to control object inclusion, see [“Using Rules to Specify Execution Timing and Control Object Inclusion” on page 352](#).

22.3 Setting Up the Objects Necessary for Live Output

As is the case with traditional Exstream output, Live output requires you to set up an output object and an output queue. These two items serve the same purpose in Exstream Live as they do in Exstream Design and Production—that is, they allow you to customize the production settings for your output. For Live output, however, an additional object is required: the Live setting object. In a sense, the Live setting object forms the basis for your Live application, because it controls the views, themes, and actions that make Live output interactive.

To set up the objects necessary for DLF output, you must complete the following tasks:

1. [“Creating a Live Setting Object” below](#)
2. [“Creating a DLF Output Object” below](#)
3. [“Creating a DLF Output Queue” on page 357](#)

You can also complete the following optional task as needed:

- [“Defining How Resources Are Handled in the Output” on page 358](#)

22.3.1 Creating a Live Setting Object

A Live setting object is always required in order to create DLF output. The Live setting object allows you to associate views, themes, and actions with a DLF file—in other words, it is what makes a Live document interactive. Additionally, it controls other options in a Live document, such as available interface options, available fonts and colors, and security options, among other things.

After you have created a Live setting object, you must then associate it with a DLF output object, which in turn is added to a DLF output queue. When you create the DLF output, the setting object (and therefore the themes, views, and actions) is applied.

To create a Live setting object:

1. In Design Manager, in the Library, go to **Environment > Interactive > Live Settings**.
2. Right-click the **Live Settings** heading and select **New Live Settings**.

The **New Live Settings** dialog box opens.

3. In the **Name** box, enter a name.
4. In the **Description** box, enter a description (optional).
5. Click **Finish**.

The new setting object opens in the Property Panel for you to define.

6. From the Menu bar, select **File > Save**.

22.3.2 Creating a DLF Output Object

A DLF output object combines all the settings that control the final appearance and format of the Live document. Before you can create a DLF output object, however, you must have created a Live setting object, which is always required in order to create Live output.

For more information about creating a Live setting object, see [“Creating a Live Setting Object” above](#).

To create a DLF output object:

1. In Design Manager, in the Library, go to **Environment > Delivery > Outputs**, and expand the **Outputs** heading.
2. Right-click the type of output object you wish to create and select **New Output**. For example, right-click **SBCS Outputs** to create a new SBCS output object.

The **New Output** dialog box opens.

3. In the **Name** box, enter a name.
4. In the **Description** box, enter a description (optional).
5. Click **Finish**.

The output object opens in the Property Panel for you to define.

6. Click the **Basic** tab.
7. In the **Model** box, enter descriptive information for the output object (optional). This information can help you differentiate output objects if you use internal identification conventions.
8. From the **Driver** drop-down list, select **Live**.
9. From the **Resolution** drop-down list, select the sharpness of the output (text and graphics) measured in dots per inch (dpi).
10. From the **Allowed** drop-down list, specify which parts of the Live document can be accessed by the engine when the DLF file is imported into another design in a fulfillment step.

To	Do this
Allow both design objects and customer data in XML format to be accessed	Select Content and data .
Allow only information and objects you can see in Designer to be accessed (customer data is not available)	Select Content only .
Allow only customer data in XML format to be accessed or allow the data from the DLF file to be used as a customer driver file	Select Data only .
Allow the DLF file to be used for fulfillment (neither design objects or customer data is available)	Select None .

11. If you want to include customer data in the DLF file (the **Customer #.xml** file in the **D** folder of the DLF file), select the **Include data** check box.

The **Data file for XML layout** box becomes active.

For information about how customer data is stored in the DLF file, see “[A Technical Look at the DLF File Structure](#)” on page 27.

12. In the **Data file for XML layout** box, do the following:

- a. Click .

The **Select Data File** dialog box opens.

- b. Browse to find the data file that provides the XML layout for the customer data XML files and click **OK**.

For more information about setting up data files, see “[Setting Up Data Files for the Live Document](#)” on page 364.

13. Do the following to specify the Live setting object you want to associate with this output object:

- a. In the **Settings** box, click .

The **Select Settings** dialog box opens.

- b. Browse to find the Live setting object you want to use and click **OK**.

For more information about creating a Live setting object, see “[Creating a Live Setting Object](#)” on page 355.

14. If revision tracking is turned on, you can highlight added text or images in your output with a black underline by doing the following:

- a. Click the **Advanced** tab.
 - b. Select the **Show blacklines** check box.

The **Show blacklines** feature shows only added text, not deleted text. For example, if you delete four paragraphs of text and add one, the only revision tracking that you see is a black underline that highlights the one added paragraph. For images, the **Show blacklines** option works only with embedded images in text boxes. Also, the **Show blacklines** feature underlines only text or images that are added in Designer. Text or images that are added by an end user in LiveEditor are designated by the revision tracking feature in LiveEditor, if that feature is enabled.

15. From the Menu bar, select **File > Save**.

22.3.3 Creating a DLF Output Queue

An output queue defines general printing options and associates an output object with those settings. Before you can create a DLF output queue, you must have created a DLF output object.

For more information about creating a DLF output object, see “[Creating a DLF Output Object](#)” on page 355.

To create an output queue:

1. In Design Manager, in the Library, go to **Environment > Delivery > Output Queues**, and expand the **Output Queues** heading.
2. Right-click the type of output queue you wish to create and select **New Output Queue**. For example, right-click **SBCS Output Queues** to create a new SBCS output queue.

The **New Output Queue** dialog box opens.

3. In the **Name** box, enter a name.
4. In the **Description** box, enter a description (optional).
5. Click **Finish**.

The output queue opens in the Property Panel for you to define.

6. At a minimum, you must carry out the following tasks to prepare the output queue to support DLF output:
 - On the **Basic** tab, in the **Output** box, specify a DLF output object.
 - On the **Basic** tab, in the **Production file** box, specify the name and path of the file that the DLF output object will create.
 - On the **Basic** tab, in the **Test file** box, specify the name and path of the file that the DLF output object will create.
7. From the Menu bar, select **File > Save**.

For more information on defining output queues, see *Creating Output* in the Exstream Design and Production documentation.

22.3.4 Defining How Resources Are Handled in the Output

To reduce the file size or to increase the print quality of a Live document, you can change the way fonts and images are handled. For example, you can increase or decrease image resolution to improve output quality or increase processing speed, respectively.

Note: PANTONE Colors are not supported in DLF output. If you use a PANTONE Color in a design for output that does not support PANTONE Colors, Exstream substitutes an equivalent color value.

For information about managing font resources in the output, see “[Controlling Font Behavior in Live Applications](#)” on page 248.

Managing Image Resources in the Output

1. In Design Manager, from the Library, drag the desired output object to the Property Panel.
2. Click the **Resource Management** tab.
3. From the **Resolution method** drop-down list, select from the following options:

To	Do this
Determine the resolution settings using the properties set on the Basic tab, System Settings , and image properties	Select Automatic .
Standardize all the resolution settings to the dpi specified in the Resolution drop-down list on the Basic tab	Select Printer resolution .
Set the print stream resolution to one-half of the driver specification. In most drivers, you experience little or no reduction in print quality, but processing time is reduced.	Select Half resolution .
Provide a specific resolution	Select Specific resolution .

4. From the Menu bar, select **File > Save**.

Chapter 23: Processing Live Documents

After an end user edits a Live document, you can return the Live document to the engine for processing. This processing might include sending the Live document to a different output type, packaging additional documents, or using the edited Live document as a template for multiple customer documents. You might use separate applications in Design Manager for creation of the Live document and for the processing or fulfillment of the edited Live document. You can also use Live engine switches to aid in processing.

Because of their unique XML-based structure and their compatibility with the engine, Live documents support many different processing methods after they have been edited by end users. The method you choose to use depends on your organization's workflow and how you plan to distribute the Live document to customers. The processing methods available for Live documents fall into two categories:

- **One-to-one processing**—Used to create Live documents that are customized for a specific customer
- **One-to-many processing (using Live documents as templates)**—Used to create multiple customer documents using one Live document as a template

Keep in mind that it is not a requirement to return edited Live documents to the engine for processing. For example, end users might print a completed Live document locally or save it as a PDF and distribute it to customers from their offices.

This chapter discusses the following topics:

- “[Using One-to-One Processing for Live Documents](#)” below
- “[Using One-to-Many Processing for Live Documents \(Live Documents as Templates\)](#)” on page 371
- “[Using Live Engine Switches During Processing](#)” on page 380

23.1 Using One-to-One Processing for Live Documents

Processing Live documents in a one-to-one scenario allows you to use a single Live document to produce one document that is targeted to a specific customer. When you use one-to-one processing, the engine processes an edited Live document and creates a customized document. This method allows end users to do extensive editing to customize a Live document

for a specific customer and then send the Live document content to a different output type or package it with additional documents for the customer mailing.

One-to-one processing



One-to-one processing is typically used in on-demand or real-time applications, such as enrollments, account openings, customer service representative (CSR)-initiated correspondence, or sales proposals. For example, suppose a CSR uses an interview page in LiveEditor to create a document containing quotes for three types of insurance for a customer. The Live document can be submitted to the engine, where additional documents containing specific policy information are appended to the Live document. The Live document and the policy information are generated as PDF output, which can then be emailed or printed and mailed to the customer.

You can implement one-to-one processing in several ways. The method you choose to use depends on your workflow, how you need to distribute the final output, and whether you will use the data in the Live document. The methods are as follows:

- **Use one Live document as a driver file**—This method allows you to send a Live document to a different type of output, add pages or documents to the Live document, and create a customized document for a particular customer. Optionally, you can access the

underlying data in the Live document.

- **Use multiple Live documents as driver files**—This method allows you to use batch processing to send multiple Live documents to a different type of output, add pages or documents to the Live documents, and create customized documents for multiple customers. Optionally, you can access the underlying data in the Live document.
- **Import the Live document into a design at run time**—This method allows you to import the Live document as you would import other types of files, such as images. You cannot access the data in the Live document; it is simply brought into the design as an image would be.

This section discusses the following topics:

- “[Using One Live Document as a Driver File](#)” below
- “[Using Multiple Live Documents as Driver Files](#)” on the next page
- “[Setting Up an Application to Use One Live Document or Multiple Live Documents as Driver Files](#)” on page 364
- “[Importing a Live Document at Run Time](#)” on page 371

23.1.1 Using One Live Document as a Driver File

Typically, you use a Live document as a driver file during an on-demand process. When the engine processes a completed Live document, the data in the Live document serves as the driver file to generate output. You use a special data file type (**Live**) to map the data used to drive the output.

In order to use a Live document as a driver file, your Design Manager environment must meet the following requirements:

- You must have licensed the InteractiveInput module and the Dynamic Content Import module.
- The application used to process the Live document must be in the same database as the application used to create the Live document.

Most of the tasks you must carry out to set up the application that will process the completed Live document are identical to tasks used in traditional Exstream processing. For example, you must set up the desired output object and any pages you want to include in the final customer document. However, you must complete a few additional tasks to set up the application that will process the Live document.

For more information about setting up an application to use a Live document as a driver file, see “[Setting Up an Application to Use One Live Document or Multiple Live Documents as Driver Files](#)” on page 364.

23.1.2 Using Multiple Live Documents as Driver Files

You might choose to use multiple Live documents as driver files instead of processing each Live document separately. For example, end users might work in many Live documents throughout the day and then submit them to the engine each night for processing. This method allows you to process Live documents in a batch environment, while still processing each Live document individually and creating customized output for each customer.

When using multiple Live documents as driver files, your Design Manager environment must meet the following requirements:

- You must have licensed the InteractiveFulfillment module, the InteractiveInput module, and the Dynamic Content Import module.
- The application used to process the Live documents must be in the same database as the application used to create the Live document.

You can use one of two methods to process multiple Live documents as driver files:

- Create a driver file for each Live document and include multiple driver files in the application used for fulfillment.
- Create only one driver file for the application used for fulfillment and use the DRIVERLISTFILE engine switch to specify a list of Live documents to be processed.

When you use the DRIVERLISTFILE switch, the engine reads the list of files during processing of the application used for fulfillment to determine each Live document to use as a driver file.

After the Live documents have been processed, you must update the list of files as needed; Design Manager does not automatically remove processed Live documents from the list or move Live documents to a new location after they have been processed. When you use this method, you can use entire Live documents as a driver file or you can use the customer XML contained in a Live document as a driver file. However, regardless of the source of the driver files, they must all have the same layout as the customer XML.

Most of the tasks you must carry out to set up the application that will process multiple Live documents are identical to tasks used in traditional Exstream processing. For example, you must set up the desired output object(s) and any pages you want to include in the final customer document. However, you must complete a few additional tasks to set up the application that will process the Live documents.

For more information about setting up an application to use multiple Live documents as driver files, see “[Setting Up an Application to Use One Live Document or Multiple Live Documents as Driver Files](#)” on the next page.

23.1.3 Setting Up an Application to Use One Live Document or Multiple Live Documents as Driver Files

Most of the tasks you must carry out to set up the application that will process the completed Live document are identical to tasks used in traditional Exstream processing. For example, you must set up the desired output object and any pages you want to include in the final customer document. Additionally, to set up the application that will process Live documents, you must complete the following tasks:

1. [“Setting Up a Placeholder Variable for the Live Document” below](#)
2. [“Setting Up Data Files for the Live Document” below](#)
3. [“Creating a Placeholder Document” on page 368](#)
4. [“Defining Application Resources for Live Document Processing” on page 379](#)

If you are using multiple Live documents as driver files and you are using the DRIVERLISTFILE switch, you must also complete the following additional task:

- [“Identifying the List of Live Documents to be Used as Driver Files” on page 370](#)

Setting Up a Placeholder Variable for the Live Document

You must create a variable that serves as a placeholder for the Live document that will be processed. The placeholder must use the following settings:

- The **Type** must be **Placeholder**.
- The **Placeholder Type** must be **Live**.
- If the Live document contains multiple pages, you must select the **Array** check box.

You can define the other variable properties as needed.

For information about defining general variable properties, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

Setting Up Data Files for the Live Document

To use a Live document as a driver file, you must use a unique data file type: **Interactive**.

If you are processing multiple Live documents as driver files, you can use one of two methods to handle the multiple documents:

- Create a driver file for each Live document and include multiple driver files in the application used for fulfillment.

- Create only one driver file for the application used for fulfillment and use the DRIVERLISTFILE engine switch to specify a list of Live documents to be processed.

For more information about using the DRIVERLISTFILE engine switch, see ["DRIVERLISTFILE" on page 440](#).

To set up a Live data file to process Live documents:

1. In Design Manager, create a new data file. The data file must have a **File type** of **Customer driver** or **Reference** and a **File format** of **Interactive**.

For information about creating data files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

Caution: If you select **Live** as the file format for a reference file, you must select **Memory** from the **Access** drop-down list on the **Basic** tab of the reference file properties.

2. Click the **Interactive** tab.
3. To define the placeholder variable you will use to bring in the Live document, complete the following steps:

- a. In the **Placeholder variable** box, click .

The **Select Variable** dialog box opens.

- b. Select the placeholder variable you created for the Live document.

For more information about creating the placeholder variable for the Live document, see ["Setting Up a Placeholder Variable for the Live Document" on the previous page](#).

- c. Click **OK**.

The **Select Variable** dialog box closes and the name of the variable appears in the **Placeholder variable** box.

4. From the **Content/data use** drop-down list, select one of the following options to define how the Live document is processed, and then complete any remaining steps.

Caution: The option you select from the **Content/data use** drop-down list must match the option you select on the **Allowed use** drop-down list of the properties of the Live output object that you use in the application used for processing or fulfillment. Otherwise, you receive a packaging error message.

To	Do this
Process only the content in the Live document (for example, to import the Live document without accessing the data it contains)	Select Content .
Process only the data in the Live document (for example, to populate other documents with data or to trigger other processes)	<ol style="list-style-type: none">a. Select Data.b. Click the Basic tab.c. From the Data mapping method drop-down list, select Schema or Sample file (manual), depending on whether you are using a schema or sample XML for mapping the data.d. In the Data mapping source box, click  . The Open dialog box opens.e. Browse to the XML sample file or schema that provides a sample layout of the Live document XML.<p>Tip: Often, this layout file is the same layout file you specify in the Data file for XML layout box in the output object properties.</p><p>Note: In a DBCS application, the XML file you use must be encoded in UTF-16 little-endian format.</p>f. Select the file or schema and click Open. The Open dialog box closes and the XML sample file or schema appears in the Data mapping source box.

To	Do this
Process both the content and the data in the Live document	<ol style="list-style-type: none">a. Select Content and Data.b. Click the Basic tab.c. From the Data mapping method drop-down list, select Schema or Sample file (manual), depending on whether you are using a schema or sample XML for mapping the data.d. In the Data mapping source box, click . The Open dialog box opens.e. Browse to the XML sample file or schema that provides a sample layout of the Live document XML. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"><p>Tip: Often, this layout file is the same layout file you specify in the Data file for XML layout box on the output properties.</p></div> <p>DBCS: The XML file you use must be encoded in UTF-16 little-endian format.</p> <ol style="list-style-type: none">f. Select the file or schema and click Open. The Open dialog box closes and the XML sample file or schema appears in the Data mapping source box.

5. On the **Test Data Source** and **Production Data Source** tabs, select the Live document that will drive the processing.

Note: If you are using multiple data files to process multiple Live documents, the files you specify on the **Test Data Source** and **Production Data Source** tabs do not have any effect on the processing. During processing, the files specified in the driver list file will serve as the data sources instead of any files specified on these tabs.

For information about using multiple Live documents as driver files, see “[Using Multiple Live Documents as Driver Files](#)” on page 363.

6. Drag the data file to the Edit Panel and map it as needed.

For information about mapping data files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

7. If you are creating multiple data files to process multiple Live documents, repeat step 1 through step 6 to create additional data files as needed.
8. From the Menu bar, select **File > Save**.

Creating a Placeholder Document

1. Create a new document.

For information about creating documents, see *Designing Customer Communications* in the Exstream Design and Production documentation.

2. Click the **Basic** tab.
3. From the **Document type** drop-down list, select **Placeholder (use pre-composed content)**.
4. To define the variable used to import the Live document, complete the following steps:

- a. In the **Placeholder variable** box, click .

The **Select Variable** dialog box opens.

- b. Select the placeholder variable you created for the Live document.

For information about creating the placeholder variable for the Live document, see [“Setting Up a Placeholder Variable for the Live Document” on page 364](#).

- c. Click **OK**.

The **Select Variable** dialog box closes and the variable appears in the **Placeholder variable** box.

- d. Define the other document properties as needed.

5. Add a page to the document to make it structurally valid. The name you assign to a page will be applied to all pages imported into that document in LiveEditor.

Tip: If you want to include additional content for the customer, add those pages to the document or to a different document as needed.

For information about adding pages and defining general document properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

6. If the document will contain multiple pages, complete the following steps to configure the page object to accept multiple pages:

- a. From the Library, drag the document to the Edit Panel.

A graphical representation of the document appears.

- b. In the right column, double-click the page name.

The **Document Page Properties** dialog box opens.

- c. From the **Position of page in document** drop-down list, select **Filler Page (as Required)**.

This option allows the page to duplicate itself so that it can accept all of the pages contained in an imported file.

- d. Click **OK**.

The **Document Page Properties** dialog box closes.

7. From the Menu bar, select **File > Save**.

For information about defining general document properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

Defining Application Resources for Live Document Processing

In the application properties of the application used for processing or fulfillment of the Live document, you must define the resources that are required in order to process the Live document correctly. For example, if fonts are not included in the Live document that will be processed, they must be packaged with the application used for processing so that the final output appears correctly. Usually, you need to include only the resources from the application used to create the original Live document, but sometimes you might need to include other resources.

To define the application resources:

1. In Design Manager, from the Library, drag the application used to process the Live document to the Property Panel.
2. Click the **Interactive** tab.
3. To define the application resources that must be available when the Live document is processed, select one of the following options from the **Resources to package for Interactive document post-processing** drop-down list and complete the steps as necessary:

To	Do this
Include all the resources used by every application in the database	Select All applications .

To	Do this
Include all the resources used by applications you specify	<ol style="list-style-type: none">a. Select Applications in list.b. Under the Interactive applications to be processed box, click  <p>The Select Application dialog box opens.</p> <ol style="list-style-type: none">c. Browse to the application that contains the resources you need to use.d. Click OK. <p>The Select Application dialog box closes and the application appears in the Interactive applications to be processed box.</p> <ol style="list-style-type: none">e. Repeat step b through step d to add applications as needed.
Include all the resources used by applications in a folder you specify	<ol style="list-style-type: none">a. Select Applications in folders in list.b. Under the Interactive applications to be processed box, click  <p>The Select Folder dialog box opens.</p> <ol style="list-style-type: none">c. Browse to the folder that contains the applications that use the resources you need to use.d. Click OK. <p>The Select Folder dialog box closes and the folder name appears in the Interactive applications to be processed box.</p> <ol style="list-style-type: none">e. Repeat step b through step d to add folders as needed.

4. Define other application properties as needed.

For information on defining general application properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

5. From the Menu bar, select **File > Save**.

Identifying the List of Live Documents to be Used as Driver Files

When you use multiple Live documents as driver files but want to use only one driver file in the application used for fulfillment, you must use the DRIVERLISTFILE engine switch in your control file to provide the location of the file that identifies the Live documents to be read as driver files. This file specifies the path to each Live document to be used as a driver file (with one path on each line).

Sample file containing list of Live documents

```
C:\Program Files\Exstream\CustomerWelcome003423.DLF
C:\Program Files\Exstream\CustomerWelcome003553.DLF
C:\Program Files\Exstream\CustomerWelcome003801.DLF
C:\Program Files\Exstream\CustomerWelcome004463.DLF
C:\Program Files\Exstream\CustomerWelcome004493.DLF
C:\Program Files\Exstream\CustomerWelcome004528.DLF
C:\Program Files\Exstream\CustomerWelcome004687.DLF
C:\Program Files\Exstream\CustomerWelcome005923.DLF
C:\Program Files\Exstream\CustomerWelcome006702.DLF
C:\Program Files\Exstream\CustomerWelcome006723.DLF
```

For more information about the DRIVERLISTFILE switch, see “[DRIVERLISTFILE](#)” on [page 440](#).

For information about using control files, see *Preparing Applications for Production* in the Exstream Design and Production documentation.

23.1.4 Importing a Live Document at Run Time

If you do not need to use the data in a Live document, you can simply import the Live document as you would import other types of files, such as images. To import a Live document into a design at run time, set up an application using a placeholder variable with a placeholder type of **Live**. You must have licensed the Dynamic Content Import module to import a Live document at run time.

For information about setting up applications to import content, see *Importing External Content* in the Exstream Design and Production documentation.

If you import multiple Live documents (in batch mode), you must use the CUSTOMER_RESET_IMPORTS engine switch. This switch ensures that the engine restarts for each customer so that the customer information in the Live documents is honored.

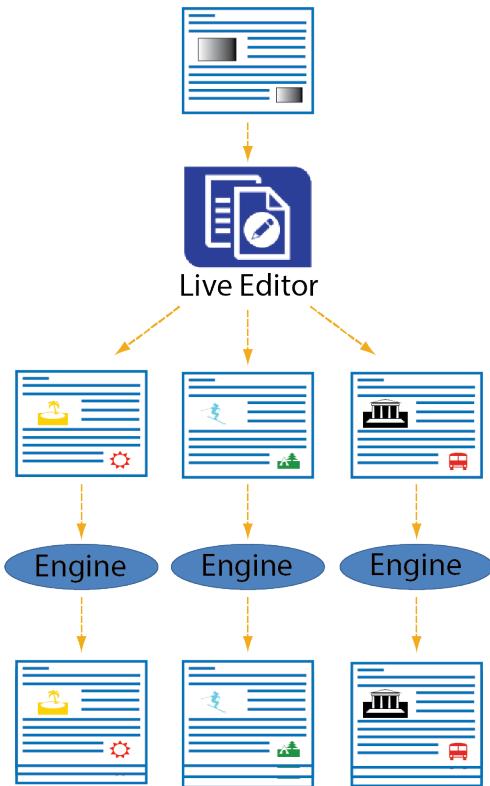
For more information about the CUSTOMER_RESET_IMPORTS switch, see *Switch Reference* in the Exstream Design and Production documentation.

23.2 Using One-to-Many Processing for Live Documents (Live Documents as Templates)

Processing Live documents in a one-to-many workflow allows you to use one or more data sources to create multiple customer documents from one Live document. This type of processing is also known as using a Live document as a template. One-to-many processing is typically used in batch-processed applications, such as sales brochures, direct mail, and corporate communications. For example, suppose you use a Live document to create a travel

brochure that is edited by end users at a regional office. You can allow end users to choose different content and images depending on their customers' interests. An end user might use the Live document to create a brochure designed for customers who enjoy cruises. Before returning the Live document to the engine for processing, the end user provides a distribution list containing a list of all the customers who should receive that brochure. During processing, the engine uses the customized Live document as a template to create a cruise brochure for each customer included in the distribution list.

One-to-many processing



To use one-to-many processing, you must have licensed the InteractiveFulfillment module and the Dynamic Content Import module.

If your Live document contains variable values that will be updated based on data contained in the fulfillment driver file, then your Live document must be a template. For example, suppose your Live document contains an existing value for an account number variable. However, your fulfillment driver file contains updated data that will change the value of that variable. In this scenario, your Live document must be a template in order for the value of the account number variable to be updated during fulfillment.

Keep in mind that if you use a Live document as a template, you must also use it as a driver file in the fulfillment run. However, because you cannot use different DLF files as driver files, you cannot set up different Live document templates to be independent driver files in a fulfillment run. Likewise, you cannot process more than one Live document template at a time during fulfillment.

You can implement one-to-many processing in two ways. The method you choose to use depends on your workflow, how you need to distribute the final output, and how you want to manage the data that drives the document creation. The methods are as follows:

- **Use a driver file maintained outside the Live document**—This method allows you to use a driver file separate from the Live document DLF file to drive the creation of new documents. For example, if you are using a large driver file, such as for a document sent to thousands of customers, you might choose to maintain the driver file outside of the Live document.
- **Use a distribution list contained in the Live document**—This method allows end users to supply a driver file (a distribution list) that is used by the engine to drive the creation of new documents.

This section discusses the following topics:

- “[Using a Driver File Maintained Outside of the Live Document](#)” below
- “[Using a Distribution List Contained in a Live Document](#)” on page 376

23.2.1 Using a Driver File Maintained Outside of the Live Document

When a driver file is large and cannot be included in a Live document easily, you can maintain the driver file outside of the Live document. For example, if you use a Live document to manage standard communications about policy renewals, and you send the Live document to thousands of customers, you might choose to maintain the customer driver file centrally, rather than requiring end users to supply a large customer driver file each time they interact with the Live document.

You must have licensed the InteractiveFulfillment module and the Dynamic Content Import module in order to use one-to-many processing.

To use a driver file maintained outside of the Live document, you must set up the application used to create the Live document so that the original Live document is designed to support this type of driver file. Next, you must set up data files and define the resources for the application used for fulfillment.

To use a Live document as a template with a distribution file maintained outside of the Live document, you must complete the following tasks:

1. [“Defining the Live Document as a Template” on the next page](#)
2. [“Setting Up to Import the Live Document When Using an External Driver File” on the next page](#)
3. [“Setting Up a Data File to Supply the Customer Information” on page 375](#)
4. [“Defining Application Resources for Live Document Processing” on page 379](#)

Defining the Live Document as a Template

In the application that creates the original Live document, you must identify the Live document as a file that can be sent back to the engine and processed to create multiple new documents.

To define the Live document as a template:

1. In Design Manager, from the Library, drag the setting object used for the Live document to the Property Panel.
2. Click the **Data** tab.
3. Select the **Live document is a template** check box.
4. Define other setting properties as needed.

For information about defining setting properties, see “[Creating a Live Setting Object](#)” on [page 355](#).

5. From the Menu bar, select **File > Save**.

Setting Up to Import the Live Document When Using an External Driver File

When you use a driver file maintained outside of the Live document to drive the creation of new documents, you import the Live document into the application used for processing in the same way you import Live documents at run time for one-to-one processing. Most of the tasks you must carry out to set up the application that will process the completed Live document are identical to tasks used in traditional Exstream processing, and you must carry out the same set of additional tasks to set up an application to use one-to-many processing as you would complete for an application that uses one-to-one processing.

Keep in mind that you must order the data files in the application so that the data file that is mapped to import the Live document comes before the data file that supplies the customer information.

To set up the application that will process Live documents, you must complete the following tasks:

1. Set up a placeholder variable for the Live document.

For more information, see “[Creating a Placeholder Document](#)” on [page 368](#).

2. Set up a data file for the Live document.

For more information, see “[Setting Up Data Files for the Live Document](#)” on [page 364](#).

Note: In the application in the Library, you must order the data files so that the data file that is mapped to import the Live document comes before the data file that supplies the customer information.

3. Create a placeholder document.

For more information, see [“Creating a Placeholder Document” on page 368](#).

Setting Up a Data File to Supply the Customer Information

Because you defined the original Live document as a template, when the engine processes the file, it also locates the data that drives the creation of new documents. This data can be in any type of data file; however, it must meet the following requirements:

- The data must match the layout required by the Live document.
- In the application, you must place this data file after the data file mapped to import the Live document.
- You can use XML generated with external systems or processes to supply the customer XML for a Live document, but to be fully supported, the structure must match the structure of XML created using a Live document. XML created from data entered in Live documents in LiveEditor is generated in the correct format automatically.

For general information about creating and mapping data files, see *Using Data to Drive an Application* in the Exstream Design and Production documentation.

Defining Application Resources for Live Document Processing

In the application properties of the application used for processing or fulfillment of the Live document, you must define the resources that are required in order to process the Live document correctly. For example, if fonts are not included in the Live document that will be processed, they must be packaged with the application used for processing so that the final output appears correctly. Usually, you need to include only the resources from the application used to create the original Live document, but sometimes you might need to include other resources.

To define the application resources:

1. In Design Manager, from the Library, drag the application used to process the Live document to the Property Panel.
2. Click the **Interactive** tab.
3. To define the application resources that must be available when the Live document is processed, select one of the following options from the **Resources to package for**

Interactive document post-processing drop-down list and complete the steps as necessary:

To	Do this
Include all the resources used by every application in the database	Select All applications .
Include all the resources used by applications you specify	<ol style="list-style-type: none">Select Applications in list.Under the Interactive applications to be processed box, click . The Select Application dialog box opens.Browse to the application that contains the resources you need to use.Click OK. The Select Application dialog box closes and the application appears in the Interactive applications to be processed box.Repeat step b through step d to add applications as needed.
Include all the resources used by applications in a folder you specify	<ol style="list-style-type: none">Select Applications in folders in list.Under the Interactive applications to be processed box, click . The Select Folder dialog box opens.Browse to the folder that contains the applications that use the resources you need to use.Click OK. The Select Folder dialog box closes and the folder name appears in the Interactive applications to be processed box.Repeat step b through step d to add folders as needed.

- Define other application properties as needed.

For information on defining general application properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

- From the Menu bar, select **File > Save**.

23.2.2 Using a Distribution List Contained in a Live Document

You can allow end users working with a Live document in LiveEditor to provide the customer information used to drive the creation of new documents from a Live document. This workflow is often used when regional or franchise offices need to make selections in a Live document and

then provide the list of customers who will receive that document. For example, suppose you use a Live document as a template for a travel brochure. End users can select the types of trips they want to include in the brochure for their groups of customers. They then upload the list of customers (called a "distribution list" in LiveEditor) for whom the customized brochure is designed. The distribution list travels with the Live document to the engine, where it is used to create documents based on the customers in the distribution list.

You must have licensed the InteractiveFulfillment module and the Dynamic Content Import module in order to use a driver file contained in a Live document.

To use a driver file contained in the Live document, you must set up the application used to create the Live document so that the original Live document is designed to support this type of driver file. Next, you must set up a data file and define the resources for the application used for fulfillment.

To use a Live document as a template with a distribution file contained in the Live document, you must complete the following steps:

1. ["Setting Up the Live Document to Include a Distribution List During Processing" below](#)
2. ["Setting Up to Import the Live Document When Using an Included Distribution List" on the next page](#)
3. ["Defining Application Resources for Live Document Processing" on page 379](#)

Setting Up the Live Document to Include a Distribution List During Processing

In the application that creates the Live document, you must identify the Live document as a file that can be sent back to the engine and processed to create multiple new documents. In addition, because the end users will provide the customer information that will drive the documentation creation, you must also define the types of driver files that are allowed. You define these properties on the setting object as follows:

1. In Design Manager, from the Library, drag the setting object used for the Live document to the Property Panel.
2. Click the **Data** tab.
3. Select the **Live document is a template** check box.
4. In the **Distribution list driver files** box, specify data files that define the layout allowed in distribution lists.

- a. Click .

The **Distribution list driver files** dialog box opens.

- b. In the **Data file** box, click .

The **Select Data File** dialog box opens.

- c. Select the data file that defines the layout and click **OK**.

The **Select Data File** dialog box closes and the data file appears in the **Data file** box.

- d. In the **Description** box, enter a description of the layout that the end user can use to identify the file.
- e. In the **File extensions** box, enter the file extensions allowed for the distribution file.
- f. Click **OK**.

The **Distribution list driver files** dialog box closes.

5. Repeat step 4 to add as many data files as needed.

6. Define other setting properties as needed.

For information about defining setting properties, see “[Creating a Live Setting Object](#)” on [page 355](#).

7. From the Menu bar, select **File > Save**.

Setting Up to Import the Live Document When Using an Included Distribution List

When you use a distribution list contained in the Live document to drive the creation of new documents, you import the Live document into the application used for processing in the same way you import Live documents at run time for one-to-one processing. Most of the tasks you must carry out to set up the application that will process the completed Live document are identical to tasks used in traditional Exstream processing, and you must carry out the same set of additional tasks to set up an application to use one-to-many processing as you would complete for an application that uses one-to-one processing.

Because you defined the original Live document as a file that will be used to create multiple documents, when the engine processes the completed Live document, it also locates the distribution list in the Live document and uses it to drive the creation of new documents.

To set up the application that will process Live documents, you must complete the following tasks:

1. Set up a placeholder variable for the Live document.

For more information, see “[Creating a Placeholder Document](#)” on [page 368](#).

2. Set up a data file for the Live document.

For more information, see “[Setting Up Data Files for the Live Document](#)” on [page 364](#).

3. Create a placeholder document.

For more information, see [“Creating a Placeholder Document” on page 368](#).

Defining Application Resources for Live Document Processing

In the application properties of the application used for processing or fulfillment of the Live document, you must define the resources that are required in order to process the Live document correctly. For example, if fonts are not included in the Live document that will be processed, they must be packaged with the application used for processing so that the final output appears correctly. Usually, you need to include only the resources from the application used to create the original Live document, but sometimes you might need to include other resources.

To define the application resources:

1. In Design Manager, from the Library, drag the application used to process the Live document to the Property Panel.
2. Click the **Interactive** tab.
3. To define the application resources that must be available when the Live document is processed, select one of the following options from the **Resources to package for Interactive document post-processing** drop-down list and complete the steps as necessary:

To	Do this
Include all the resources used by every application in the database	Select All applications .
Include all the resources used by applications you specify	<ol style="list-style-type: none">a. Select Applications in list.b. Under the Interactive applications to be processed box, click The Select Application dialog box opens.c. Browse to the application that contains the resources you need to use.d. Click OK.The Select Application dialog box closes and the application appears in the Interactive applications to be processed box.e. Repeat step b through step d to add applications as needed.

To	Do this
Include all the resources used by applications in a folder you specify	<ol style="list-style-type: none">a. Select Applications in folders in list.b. Under the Interactive applications to be processed box, click The Select Folder dialog box opens.c. Browse to the folder that contains the applications that use the resources you need to use.d. Click OK.The Select Folder dialog box closes and the folder name appears in the Interactive applications to be processed box.e. Repeat step b through step d to add folders as needed.

4. Define other application properties as needed.

For information on defining general application properties, see *Designing Customer Communications* in the Exstream Design and Production documentation.

5. From the Menu bar, select **File > Save**.

23.3 Using Live Engine Switches During Processing

To customize a Live application during engine processing, you can use engine switches that are specific to processing Live documents from the command prompt or from a control file. While you can use many of the Exstream engine switches when you create Live documents, the following switches are specific to the Live environment:

- DISABLE_EMBED_DATA
- DLFCAMPDISABLE
- DRIVERLISTFILE
- PROMOTECAMPTODLF
- RETAIN_VARIABLE_RESET_TIME
- SHOWREVISION
- USE_DLF_STYLE

For more information about Live processing switches, see “[Live Processing Switches](#)” on page 438.

Appendix A: Live Built-In Functions

Like the built-in functions available as part of the Exstream Design and Production environment, Live built-in functions contain pre-created logic that can be combined with logic that you specify to perform specific actions. Specifically, you can use Live built-in functions to perform a variety of actions in LiveEditor. For example, suppose you are designing a Live document to be used by insurance agents to send general policy information to their customers, including boilerplate information about each policy type. You can set up a selection list that allows agents to specify the types of policies held by each customer. You can then associate the values in the selection list with the `LiveExternalDocImport` built-in function so that each policy type receives the appropriate boilerplate information.

This appendix describes the following Live-specific built-in functions that you can use in Live documents:

" <code>LiveAction</code> " on the next page	" <code>LiveOpen</code> " on page 406
" <code>LiveBalloon</code> " on page 383	" <code>LivePrint</code> " on page 407
" <code>LiveBeep</code> " on page 384	" <code>LiveSave</code> " on page 408
" <code>LiveCheck</code> " on page 385	" <code>LiveSaveAs</code> " on page 409
" <code>LiveClose</code> " on page 386	" <code>LiveSaveAsPDF</code> " on page 410
" <code>LiveDataFileClose</code> " on page 388	" <code>LiveSelection</code> " on page 413
" <code>LiveDataFileOpen</code> " on page 388	" <code>LiveSendEmail</code> " on page 414
" <code>LiveDebugMessage</code> " on page 390	" <code>LiveSetPrivacyMask</code> " on page 416
" <code>LiveEnvironment</code> " on page 391	" <code>LiveSetReadOnly</code> " on page 417
" <code>LiveEraseSignature</code> " on page 393	" <code>LiveSetStyleSheet</code> " on page 418
" <code>LiveExternalDocImport</code> " on page 394	" <code>LiveSign</code> " on page 419
" <code>LiveGetSignature</code> " on page 395	" <code>LiveStore</code> " on page 420
" <code>LiveGetSignatureByIndex</code> " on page 397	" <code>LiveSubmit</code> " on page 421
" <code>LiveGetSignatureCount</code> " on page 398	" <code>LiveTestMembership</code> " on page 421
" <code>LiveGoTo</code> " on page 399	" <code>LiveValidate</code> " on page 422
" <code>LiveLaunchURL</code> " on page 401	" <code>LiveView</code> " on page 423
" <code>LiveMessage</code> " on page 403	" <code>LiveXMLRead</code> " on page 424
" <code>LiveNavigate</code> " on page 405	" <code>LiveXMLWrite</code> " on page 425

For general information about built-in functions or information about other types of built-in functions, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

A.1 LiveAction

The `LiveAction` function performs the action you specify, just as if you selected the action from the menu. Actions allow you to make custom activities easier by building them into a Live document so that they are available to end users in LiveEditor. For example, you can make it easier for end users to contact your support team by creating an action to open an email program, supply the address for your support team, and attach the Live document being edited. After you define an action, it will be available from the LiveEditor Menu bar.

For more information about creating actions, see “[Using Actions to Automate Editing Tasks](#)” on page 129.

A.1.1 Syntax

`LiveAction(EditActionName)`

A.1.2 Argument

AddAction argument

Argument	Data type	Use	Description
EditActionName	String	Required	The action to be performed. The action name must appear in quotation marks.

A.1.3 Return Codes

`LiveAction` returns an integer specifying the results of the requested operation. The following table lists the possible returns.

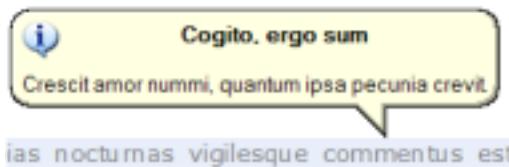
LiveAction return codes

Return code	Description
0	Success
1	Specified design object not found.
66	Circular loop in Live actions detected—exiting current action.
97	LiveEditor is required for this operation.

A.2 LiveBalloon

The `LiveBalloon` function allows you to display help in the form of a balloon that appears when end users hover the pointer over a specified interactive area.

Information balloon in LiveEditor



A.2.1 Syntax

```
LiveBalloon(Title, Content[, AnchorPoint, Icon, URL])
```

A.2.2 Arguments

LiveBalloon arguments

Argument	Data type	Use	Description
Title	String	Required	A string of text describing the object or text area that appears in bold at the top of the balloon. The text must appear in quotation marks.
Content	String	Required	A string of text that appears below the value of <code>Title</code> in the balloon. The text must appear in quotation marks.
AnchorPoint	Integer	Optional	An integer representing the anchor point. This argument takes one value (0), which is the active object or text area.
Icon	Integer	Optional	An integer representing the type of icon. The default is 0. See the table that follows for an explanation of the icons available for use with the <code>LiveBalloon</code> built-in function.
URL	String	Optional	A hyperlink. When an end user clicks the balloon, the specified webpage opens. The hyperlink must appear in quotation marks.

Icon Arguments

Specify the integer that corresponds with the type of icon that will appear in the balloon help. The following table outlines the available icons.

Icon arguments

Icon integer	Description	Appears as
0	None	No icon
1	Exclamation	
2	Information	
3	Question	
4	Error	
5	Application	
6	Secure/Lock	
7	Stop	

A.3 LiveBeep

When it is associated with a particular action, the `LiveBeep` function plays the Windows sound you specify. For example, if you want to alert end users to submit a Live document before closing it, you can write logic that triggers a warning beep when they attempt to close the Live document without first submitting it for processing.

A.3.1 Syntax

`LiveBeep([Type])`

A.3.2 Argument

LiveBeep argument

Argument	Data type	Use	Description
Type	Integer	Optional	An integer representing the system sound. The default is 0.

Type Arguments

Specify the integer that corresponds with the type of sound to be played. The following table outlines the available sounds.

Type arguments

Type integer	Description
0	Simple—A simple keyboard/speaker beep
1	Warning—Windows exclamation
2	Information—Windows asterisk
3	Question—Windows question
4	Error—Windows critical stop
5	Default—Windows default beep

A.4 LiveCheck

The LiveCheck function checks your document for spelling and grammatical errors, and for excluded words (if your company has an excluded words list).

You can use the LiveCheck function to require users to check all or part of a Live document for spelling and grammar errors, as well as excluded words. For example, suppose you create a custom function called RequiredSpellCheck that uses the LiveCheck function as part of its logic, which requires that end users check the entire Live document for spelling errors. You can then assign the RequiredSpellCheck function to a LiveEditor event—such as submitting the DLF file for processing—that triggers your custom function. The **Submit** button will not become available to end users until they have checked the entire Live document for spelling errors.

For more information about assigning LiveEditor events, see “[Executing Functions During LiveEditor Events](#)” on page 132.

A.4.1 Syntax

`LiveCheck([Interactive, "ErrorType", "Scope"])`

A.4.2 Arguments

LiveCheck arguments

Argument	Data type	Use	Description
Interactive	Boolean	Optional	<p>Controls the Spelling, Grammar, & Excluded Words dialog box. One of the following values is returned:</p> <ul style="list-style-type: none">• TRUE—Opens the dialog box. This value is the default.• FALSE—Turns on the options specified in Arguments, but does not open the dialog box
ErrorType	String	Optional	<p>Identifies the types of errors to check. Supply one of the following values:</p> <ul style="list-style-type: none">• Excluded—Turns on excluded word check and checks the Live document for words you should exclude• Grammar—Turns on grammar check and checks the Live document for grammatical errors• Spelling—Turns on spell check and checks the Live document for spelling errors <p>To specify more than one type of error check in the ErrorType argument, separate the types with a delimiter.</p> <p>For example:</p> <pre>LiveCheck(false, "Grammar,Spelling")</pre>
Scope	String	Optional	<p>Determines the scope of the spelling, grammar, and/or excluded word check. Supply one of the following values:</p> <ul style="list-style-type: none">• Page—Checks the current page• Document—Checks the current document• Customer—Checks all documents for a particular customer• File—Checks the entire Live document. This value is the default.

A.5 LiveClose

The **LiveClose** function closes the current Live document without saving it, and optionally navigates to a new URL. If you are using the **LiveClose** function in addition to other functions, **LiveClose** should appear at the end of the list. **LiveClose** causes the current function and all calling functions to exit immediately. For example, you could associate **LiveSave** and **LiveClose** with a submit button so that when end users submit a Live document for processing, LiveEditor automatically saves and closes the file.

A.5.1 Syntax

```
ReturnStatus = LiveClose([PromptIfChanged, NewURL, ForceRefresh,  
LiveEditorClose])
```

A.5.2 Arguments

LiveClose arguments

Argument	Data type	Use	Description
PromptIfChanged	Boolean	Optional	The Boolean value specifying whether the document has changed. If the value is TRUE and the document has been changed, LiveEditor issues a prompt to cancel the close so changes are not lost. The default value is FALSE.
NewURL	String	Optional	A hyperlink. If you provide a URL, the web browser navigates to the indicated URL after the Live document closes. In order for the provided URL to open, the Live document must already be open in Internet Explorer.
ForceRefresh	Boolean	Optional	The web browser bypasses the cache, forcing a page refresh. The default is TRUE.
LiveEditorClose	Boolean	Optional	The Live document is closed, and then LiveEditor is closed. This argument is executed after all other LiveClose arguments. If other Live documents are open in LiveEditor, LiveEditor prompts the end user to save them before closing. Note: If the Live document is open in a browser window, LiveClose closes the LiveEditor open in the browser but does not close the browser.

A.5.3 Return Codes

LiveClose returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveClose return codes

Return code	Description
-1	End user cancelled the operation.
0	Success
97	LiveEditor is required for this operation.

A.6 LiveDataFileClose

The `LiveDataFileClose` function closes a data file when you are finished reading or writing data. LiveEditor automatically closes any open data files when you close a DLF file. However, you can use the `LiveDataFileClose` function to close the data file without having to close the DLF file.

A.6.1 Syntax

`LiveDataFileClose(DataFileName)`

A.6.2 Argument

`LiveDataFileClose` argument

Argument	Data type	Use	Description
<code>DataFileName</code>	String	Required	The name of the Exstream data file to be closed. This name must appear in quotation marks.

A.6.3 Return Codes

`LiveDataFileClose` returns an integer specifying the results of the requested I/O operation. The following table lists the possible returns.

`LiveDataFileClose` return codes

Return code	Description
0	Success
0	Specified design object not found.
97	LiveEditor is required for this operation.

A.7 LiveDataFileOpen

The `LiveDataFileOpen` function opens a **File** dialog box, using the specified extension and description to optionally override those specified in the data file properties. You can use the

LiveDataFileOpen function to trigger reading or writing data to a data file, or you can use it to ensure that a data file is open if you want to read or write data to the file at a later point in time.

A.7.1 Syntax

LiveDataFileOpen(DataFileName[, Trigger, Extension, Description])

A.7.2 Arguments

Note: If the Extension and Description arguments are defined for the data file, then triggering the file prompts the end user to select the file to be opened. After the file is opened, it follows the instructions defined by the trigger.

LiveDataFileOpen arguments

Argument	Data type	Use	Description
DataFileName	String	Required	The name of the Exstream data file to be opened. This name must appear in quotation marks.
Trigger	Boolean	Optional	If the value of Trigger is TRUE, then the default operation for the data file is performed: <ul style="list-style-type: none">• Customer driver or Initialization file—Entire file is read.• Reference file—Lookup is performed.• Report file—Write is performed.• If the value of Trigger is FALSE, then you must use the Trigger function to read or write the data. For more information about the Trigger function, see <i>Using Logic to Drive an Application</i> in the Exstream Design and Production documentation.
Extension	String	Optional	The default extension of the file to appear in the File dialog box. This extension must appear in quotation marks.
Description	String	Optional	A description of the file, such as the type of file to appear in the File dialog box. This description must appear in quotation marks.

A.7.3 Return Codes

LiveDataFileOpen returns an integer specifying the results of the requested I/O operation. The following table lists the possible returns.

Note: In addition to these return codes, `LiveDataFileOpen` also returns Trigger return codes.

For more information about the Trigger function, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

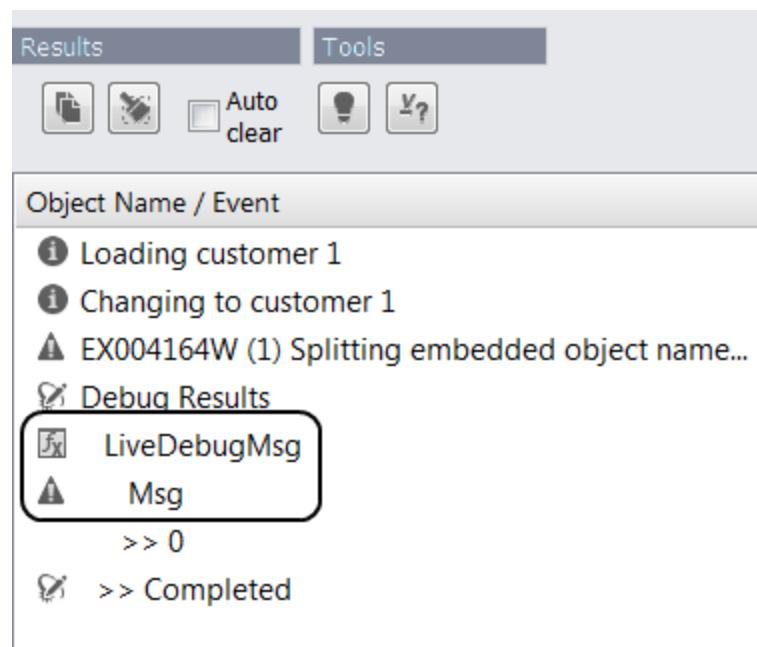
LiveDataFileOpen return codes

Return code	Description
-1	End user cancelled the operation.
0	Success
22	No start customer layout found.
97	LiveEditor is required for this operation.

A.8 LiveDebugMessage

The `LiveDebugMessage` function causes a message to appear in the Live Debugger result list. You can use the `LiveDebugMessage` function to display specific messages and/or icons in the Debugger result list.

Message in Live Debugger results



A.8.1 Syntax

`LiveDebugMessage(Message[, Icon])`

A.8.2 Arguments

LiveDebugMessage arguments

Argument	Data type	Use	Description
Message	String	Required	The expression to show in the dialog box. The text must appear in quotation marks.
Icon	Integer	Optional	An integer representing the type of icon. The default is 1.

Icon Arguments

Specify the integer that corresponds with the type of icon that appears. The following table outlines the icons available.

Note: If the Debugger is in **Single stepping** mode, it pauses after the message is shown. If the Debugger is in **Run until watch** mode, it pauses (after the message is shown) only if the message is an error message (Icon 4).

Icon arguments

Icon integer	Description	Appears as
1	Warning	
2	Information	
3	Question	
4	Error	

A.9 LiveEnvironment

The `LiveEnvironment` function returns a string containing information about the LiveEditor working environment. You can use the `LiveEnvironment` function to return information such as

whether LiveEditor is running in a browser session, or whether a Live document is open in read-only mode.

A.9.1 Syntax

`LiveEnvironment(EnvironmentItem)`

A.9.2 Argument

LiveEnvironment argument

Argument	Data type	Use	Description
EnvironmentItem	String	Required	The name of the object

EnvironmentItem Arguments

Specify the string that describes the LiveEditor environment item. The following table lists the available strings.

EnvironmentItem arguments

EnvironmentItem string	Returns
ExcludedErrors	The number of excluded word errors on the page. If the excluded word error check is turned off, the return value will not be correct.
GrammarErrors	The number of grammar errors on the page. If grammar check is turned off, the return value will not be correct.
InBrowser	One of the following values is returned: "T"—In browser "F"—Not in browser
IsAuthorizationDefined	One of the following values is returned: "T"—Authorization is defined in the setting object used when the Live document was created. "F"—Authorization is not defined in the setting object used when the Live document was created.
WorkingOffline	One of the following values is returned: "T"—Not in browser and working offline "F"—In browser and working online

EnvironmentItem arguments, continued

EnvironmentItem string	Returns
LiveMode	One of the following values is returned: Blank—Not Live "RW"—Live document open in read/write mode "RO"—Live document open in read-only mode
ProductModel	One of the following values is returned: VIEWER EDITOR
SpellingErrors	The number of spelling errors on the page. If spell check is turned off, the return value will not be correct.

A.10 LiveEraseSignature

The `LiveEraseSignature` function erases the electronic signature with the ID you specify in the argument. Electronic signatures usually lock a document to prevent editing; therefore, if you want end users to be able to remove a signature so they can edit, they must be able to erase the signature, as well.

Note: In order for this function to work, you must clear the **Locked on sign** check box on the **Button** tab for each signature button you want to erase.

For more information about signature buttons, see “[Using Signature Buttons to Allow Certain End Users to Prevent Changes in Live Documents](#)” on page 311.

A.10.1 Syntax

`LiveEraseSignature(Signature ID)`

A.10.2 Argument

LiveEraseSignature argument

Argument	Data type	Use	Description
Signature ID	Integer	Required	The ID of the electronic signature. This value is usually the value of 'SYSLD_SignatureID'.

A.11 LiveExternalDocImport

The `LiveExternalDocImport` function allows external DOCX, DXF, PDF, or TIFF files to be imported automatically as one or more pages in a placeholder document within a Live document. This function dynamically populates the placeholder document with content from the external file and requires no action on the part of LiveEditor users to place the external file.

For more information about importing external DOCX, DXF, PDF, or TIFF files into a Live document, see [“Importing Pages from External Files into a Live Document” on page 166](#).

A.11.1 Syntax

`LiveExternalDocImport(Placeholder Var, FilePath, Append, Insert Index)`

A.11.2 Arguments

LiveExternalDocImport arguments

Argument	Data type	Use	Description
Placeholder Var	String	Required	The name of an existing placeholder variable being used elsewhere in the application.
FilePath	String	Required	The path to the location of the external file to be imported. The path and file name must appear in quotation marks.
Append	Integer	Optional	This argument determines whether an existing document is to be updated (replaced with new content) or added to (the imported file is appended to the end of the document). Possible values: <ul style="list-style-type: none">• 1 = Update• 2 = Add

LiveExternalDocImport arguments, continued

Argument	Data type	Use	Description
Insert Index	Integer	Optional	<p>This argument controls the placement of the new placeholder document within the Live document. Possible values:</p> <ul style="list-style-type: none">• Any positive value indicates the document in the Live document after which the placeholder document will be placed. For example, a value of 2 means that the placeholder document will be the third document for a particular customer in the Live document.• Any value greater than the number of documents in the Live document means that the placeholder document will be the last document for a particular customer in the Live document.• A value of -1 means that the placeholder document will be the last document for a particular customer in the Live document.• A value of 0 means that the placeholder document will be the first document for a particular customer in the Live document.

A.12 LiveGetSignature

The `LiveGetSignature` function returns information about the electronic signature with the ID you specify in the argument.

A.12.1 Syntax

```
LiveGetSignature(Signature ID[, "Item", "Separator"])
```

A.12.2 Arguments

LiveGetSignature arguments

Argument	Data type	Use	Description
Signature ID	Integer	Required	The ID of the electronic signature. This value is usually the value of 'SYSLD_SignatureID'.
Item	String	Optional	One or more of the item arguments, separated by the specified separator. If you leave the Item and Separator arguments blank (the default), the function returns all the information specified by the Item argument.
Separator	String	Optional	The string you want to use to separate items returned by the function. Also used when defining the items to return. Semicolons are used by default.

Item Arguments

Specify the string that describes the type of information you want. The following table lists the strings available in the default order.

Item arguments

Item string	Returns
ID	The electronic signature ID
Name	The name of the signature as specified in the Name box on the Advanced properties dialog box
Scope	The scope within the file that was locked by the signature. One of the following values is returned: <ul style="list-style-type: none">• File—Signature locked the entire file.• Customer—Signature locked the customer.• Document—Signature locked the document.• Page—Signature locked the page.• None—Signature did not lock anything.
Requirement	Whether or not the signature is required
Status	The status of the signature. One of the following values is returned: <ul style="list-style-type: none">• Optional• Required• Signed• Unsigned• Erased
Signer	The user name when the signature was created
Timestamp	The timestamp in SOAP format (yyyy-mm-dd hh:nn:ss)
Annotation	The target of the signature. This value is set when a Live document is signed.

A.12.3 Example

You use this function when you need to check the state of a signature before performing further processing in a function. For example:

```
If LiveGetSignature(SYSLD_SignatureID, "Status") = "Signed" then
    Exit function
endif
```

A.13 LiveGetSignatureByIndex

The `LiveGetSignatureByIndex` function lets you use electronic signatures, or information about electronic signatures, in other functions.

A.13.1 Syntax

`LiveGetSignatureByIndex(Index[, "Item", "Separator", "Name", "Status"])`

A.13.2 Arguments

LiveGetSignatureByIndex arguments

Argument	Data type	Use	Description
<code>Index</code>	Integer	Required	The ID of the electronic signature
<code>Item</code>	String	Optional	<p>One or more of the item arguments, separated by the specified separator. If you leave the <code>Item</code> and <code>Separator</code> arguments blank (the default), the function returns all the information specified by the <code>Item</code> argument.</p> <p>For more information about the <code>Item</code> argument, see Item Arguments in "LiveGetSignature" on page 395.</p>
<code>Separator</code>	String	Optional	The string you want to use to separate items returned by the function. Also used when defining the items to return. Semicolons are used by default.
<code>Name</code>	String	Optional	The name of the signature as specified in the <code>Name</code> box on the Advanced properties dialog box
<code>Status</code>	String	Optional	<p>The status of the signature. One of the following values is returned:</p> <ul style="list-style-type: none">• <code>Optional</code>• <code>Required</code>• <code>Signed</code>• <code>Unsigned</code>• <code>Erased</code>

A.13.3 Example

You can use the `LiveGetSignatureByIndex` function to perform an action based on the state of electronic signatures within a Live document. For example, you can create a function to list all unsigned signatures, and, if any are found, prevent the Live document from closing.

```
Dim iCount as integer, i as integer, strOut as string, Boolean bOutput
iCount = LiveGetSignatureCount("*")
for I = 1 to iCount
    if LiveGetSignatureByIndex(I, "Status", "", "") <> "Signed" then
        if not bOutput then
            strOut = strOut + chr(13) + chr(10)
        endif
        strOut = strOut & LiveGetSignature(I, "Name") & "must be signed."
    endif
next i
if length(strOut) > 0 then
    LiveMessage(strOut)
    Value = FALSE
Else
    Value = TRUE
endif
```

A.14 LiveGetSignatureCount

The `LiveGetSignatureCount` function returns the number of signatures that match the specified name, status, or both.

A.14.1 Syntax

`LiveGetSignatureCount("Name", "Status")`

A.14.2 Arguments

`LiveGetSignatureByIndex` arguments

Argument	Data type	Use	Description
Name	String	Optional	<p>The name of the signature as specified in the Name box on the Advanced properties dialog box</p> <p>Tip: You can enter "" to count the number of unnamed signatures.</p>

LiveGetSignatureByIndex arguments, continued

Argument	Data type	Use	Description
Status	String	Optional	The status of the signature. One of the following values is returned: <ul style="list-style-type: none">• Optional• Required• Signed• Unsigned• Erased

A.14.3 Example

You use this function when you need to check the number of unnamed signatures in a Live document before performing further processing in a function.

For example:

```
If LiveGetSignatureCount(" ", "Status") = "Signed" then
    Exit function
endif
```

A.15 LiveGoTo

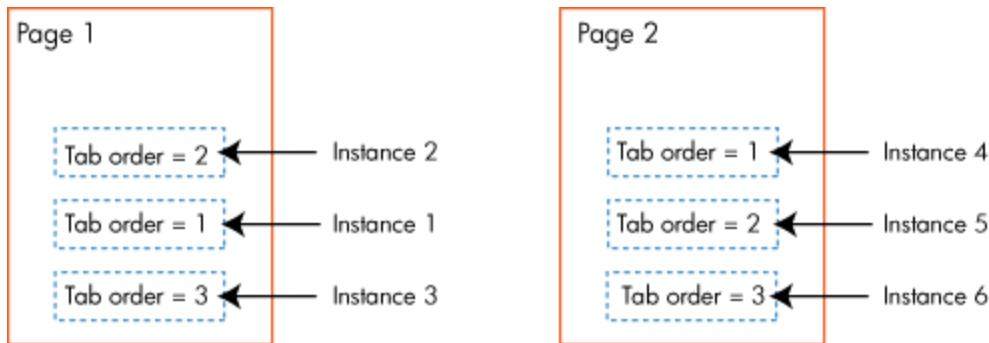
The LiveGoto function allows you to set up targeted navigation to specific named editable areas within a customer. For example, if end users select a check box to indicate that a customer lives in the United States, you can set up the Live document so that the cursor immediately moves to a page containing selections that must be made for United States' citizens.

Tip: You can use the LiveNavigate function to set up more basic navigation that spans the entire Live document (for example, navigation to the next customer, document, or page).

For more information about determining the best method for adding automated navigation to your Live document, see [“Adding Automatic Navigation to Specific Interactive Areas” on page 242](#).

When you set up the function, you specify the name of the destination area. If more than one area with the same name exists in the customer (for example, if the object can be duplicated), you must specify the instance of the area to which you want the navigation to go. If more than one area with the same name exists on the same page, the tab order of the area identifies the instance of the area. The following example illustrates these concepts. In this example, it is assumed that each interactive area (indicated by dashed lines) has the same name.

Illustration of tab order effect on instance number



In Page 2, the instance numbering continues from Page 1, since the same interactive area exists multiple times in the same customer. Therefore, to navigate to the second interactive area in Page 2, you would specify Instance 5 in the `LiveGoto` function. In Page 1, since the same interactive area exists more than once, the tab orders of the areas determine the instance number, not the order of the object on the page.

For information about the tab order of objects in a Live document, see “[Customizing the Order of Navigation Between Editable Areas](#)” on page 237.

The destination interactive area must be a tab stop in order to be found by the `LiveGoto` function. To make an interactive area a tab stop, you must select the **Tab stop** check box on the **Interactive** tab of the interactive area properties.

A.15.1 Syntax

`LiveGoto(Target[, InstanceNumber])`

A.15.2 Arguments

LiveGoTo arguments

Argument	Data type	Use	Description
Target	String	Required	<p>The name of the area where you want the cursor to be placed. This name is the name supplied in the Name box on the Advanced properties dialog box (accessed through the object properties dialog box). This box cannot be empty; you must specify an area name in order to use the <code>LiveGoto</code> function.</p> <p>Tip: You access this dialog box by clicking  on the object's properties dialog box.</p>

LiveGoTo arguments, continued

Argument	Data type	Use	Description
InstanceNumber	Integer	Optional	<p>The instance of the target area. This argument is required only if multiple objects exist with the same name (for example, if the object can be duplicated). The first instance of the area is considered to be 1. If you do not specify the InstanceNumber and multiple instances of the area exist in the customer, then the navigation automatically goes to the first instance.</p> <p>For information about determining the instance of a target area, see "Adding Automatic Navigation to Specific Interactive Areas" on page 242.</p>

A.15.3 Return Codes

LiveGoto returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveGoTo return codes

Return code	Description
0	Success
68	The instance number provided is invalid (for example, a negative value).
69	Target is hidden.
70	Target not found.

A.16 LiveLaunchURL

The LiveLaunchURL function launches a new browser or browser tab pointed at the specified URL. You might use this function to launch a custom help page or other reference material from within the Live document.

A.16.1 Syntax

`LiveLaunchURL(URL[, ForceRefresh, Flags])`

A.16.2 Arguments

LiveLaunchURL arguments

Argument	Data type	Use	Description
URL	String	Required	A hyperlink
ForceRefresh	Boolean	Optional	The web browser bypasses the cache, forcing a page refresh. The default is TRUE.
Flags	Integer	Optional	Where to launch the new URL: a new window or tab. The default is 0.

URL Argument

In addition to specific hyperlinks, you can use special arguments for the URL. They let you to go back a page or to your home page

URL arguments

URL string	Description
BACK	Takes you to the previous page. Behaves like clicking the "back" button in the web browser. Not case-sensitive.
HOME	Takes you to your home page. Behaves like clicking the "home" button in the web browser. Not case-sensitive.

Flags Argument

Specify the integer that corresponds with the location where you want to launch the new URL.

Note: If you are not running the LiveEditor in a browser, Flag is considered 0.

Flags arguments

Flags integer	Type of launch
0	Launch web browser in a new window. This argument is the default.
1	Launch web browser in a new tab. Available in Internet Explorer 7 and later only.
2	Launch web browser in a new background tab. Available in Internet Explorer 7 and later only.

A.17 LiveMessage

The `LiveMessage` function opens a dialog box to which end users must respond before they can continue.

A.17.1 Syntax

`LiveMessage(Message[, Type, Icon, Default])`

A.17.2 Arguments

`LiveMessage` arguments

Argument	Data type	Use	Description
Message	String	Required	The expression that appears in the dialog box. The text must appear in quotation marks.
Type	Integer	Optional	An integer representing the type of dialog box. The default is 0.
Icon	Integer	Optional	An integer representing the type of icon. The default is 1.
Default	Integer	Optional	An integer representing the button in left to right order that is the default. For example, if your message has two buttons (<code>OK</code> and <code>Cancel</code> , from left to right) and you specify 1 as the default, the <code>OK</code> button is highlighted by default. You can specify 1, 2, or 3. The default is 1.

Type Argument

Specify the integer that corresponds with the type of dialog box that opens. The following table outlines the types available.

Type arguments

Type integer	Dialog box displays
0	One option (<code>OK</code>)
1	Two options (<code>OK</code> and <code>Cancel</code>)
2	Three options (<code>Abort</code> , <code>Retry</code> , and <code>Ignore</code>)
3	Three options (<code>Yes</code> , <code>No</code> , and <code>Cancel</code>)
4	Two options (<code>Yes</code> and <code>No</code>)

Type arguments, continued

Type integer	Dialog box displays
5	Two options (Retry and Cancel)

Icon Argument

Specify the integer that corresponds with the type of icon that appears. The following table outlines the icons available.

Icon arguments

Icon integer	Description	Appears as
1	Warning	
2	Information	
3	Question	
4	Error	

A.17.3 Return Codes

LiveMessage returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveMessage return codes

Return code	Button Clicked
101	OK
102	Cancel
103	Abort
104	Retry
105	Ignore
106	Yes
107	No

A.18 LiveNavigate

The `LiveNavigate` function allows you to set up basic navigation to general areas in a DLF, such as navigation to a document or page in relation to the current cursor position. For example, you can add buttons to the bottom of a page to allow end users to go to the next or previous page.

Tip: You can use the `LiveGoto` function to set up targeted navigation capability within a customer (for example, navigation to a specific named interactive area).

For more information about determining the best method for adding automated navigation to your Live document, see “[Adding Automatic Navigation to Specific Interactive Areas](#)” on page 242.

A.18.1 Syntax

`LiveNavigate(Unit, Target)`

A.18.2 Arguments

LiveNavigate arguments

Argument	Data type	Use	Description
Unit	String	Required	A string representing the type of object to which you want to navigate
Target	Integer	Required	An integer representing the location of the Unit

Unit Arguments

Specify the string that corresponds with the type of object to which you want to navigate. The following table outlines the strings available.

Unit arguments

Unit string	Navigates to
C or CUSTOMER	Customer
D or DOCUMENT	Document
P or PAGE	Page
A or AREA	Object or text areas

Unit arguments, continued

Unit string	Navigates to
E or ERROR	Object or text areas with errors

Target Arguments

Specify the integer that corresponds with the location of the object relative to the current position. The following table outlines the integers available.

Target arguments

Target integer	Navigates to
0	Next
-1	Previous
-2	First
-3	Last
Any positive integer	The corresponding unit. For example, if the Target is 4 and the Unit is Customer, you navigate to the fourth customer. You can use this Target argument only with Customer, Document, or Page.

A.19 LiveOpen

The LiveOpen function opens the specified DLF file in a new window. You might use this to launch a separate Live document from within a Live document. For example, suppose you are designing a loan document that contains an option for a payment protection plan. You can associate the LiveOpen function with a submit button so that if end users select the option to add a payment protection plan to the loan, a separate payment protection form opens when they submit the Live document for processing.

A.19.1 Syntax

LiveOpen(DlfFileName)

A.19.2 Argument

LiveOpen argument

Argument	Data type	Use	Description
D1ffFileName	String	Required	The default Live document to open. The path and file name must appear in quotation marks.

A.19.3 Return Codes

LiveOpen returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveOpen return codes

Return code	Description
0	Success
30	Specified file cannot be found.
31	Specified file is already open.

A.20 LivePrint

The **LivePrint** function opens the **Print** dialog box so that the Live document can be printed. You can specify arguments that will pre-populate the **Print** dialog box with specified print settings, such as a specific document or a range of pages to print. For example, if you add a "Print this document" button to the Live document, you can use the **LivePrint** function to open the **Print** dialog box. If you specified the current document as the document to print in the function, the **Print** dialog box will be set up automatically to print the current document. End users can adjust the print settings as needed (for example, they can specify how many copies they want), and then click **Print** to print the current document.

For more information about controlling how end users can print Live documents, see ["Controlling How End Users Save and Print Live Documents" on page 261](#).

A.20.1 Syntax

`LivePrint(Unit, Range)`

A.20.2 Arguments

If you do not supply any arguments, the **Print** dialog box opens when the function is executed, but no settings are defined on it.

LivePrint arguments

Argument	Data type	Use	Description
Unit	String	Optional	<p>The unit you want the Print dialog box to be set up to print. You can specify one of the following values:</p> <ul style="list-style-type: none">• C or CUSTOMER• D or DOCUMENT• P or PAGE
Range	String	Optional	<p>The range of units to be printed (for example, you can allow end users to print documents 1 through 5). The range you specify for a page or document indicates the unit's location in the file (not in the current customer). For example, if you specify 5 as the range of the document to be printed, the fifth document in the file is printed, not the fifth document in the customer.</p> <p>You can specify ranges in any of the following ways:</p> <ul style="list-style-type: none">• Serial ranges—Use a dash to indicate the range (for example, 1-5).• Non-serial ranges—Use a comma to indicate the ranges (for example, 5, 10).• A combination—Use dashes and commas to indicate the ranges (for example, 1-5, 10, 12-30). <p>If you do not specify a value, all the pages in the specified unit will be printed.</p>

For example, you can use the following function to set up the **Print** dialog box to print all of the pages for the first customer in the Live document:

```
LivePrint ("C", "1")
```

You can use the following function to set up the **Print** dialog box to print pages 1–10 in the file:

```
LivePrint("P", "1-10")
```

A.21 LiveSave

The **LiveSave** function saves the current Live document. You can use the **LiveSave** function at certain points in the document workflow to ensure that changes are not lost, or you can combine it with another function, such as **LiveClose**. For example, you could associate **LiveSave** and

with a submit button so that when end users submit a Live document for processing, LiveEditor automatically saves and closes the file.

A.21.1 Syntax

`LiveSave(PromptIfInvalid)`

A.21.2 Argument

LiveSave argument

Argument	Data type	Use	Description
PromptIfInvalid	Boolean	Optional	The Boolean value specifying whether to prompt the end user if the variable validation finds an error. If the value is TRUE, then LiveEditor prompts the end user to fix an invalid variable value before the save is completed. The default is FALSE.

A.21.3 Return Codes

LiveSave returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveSave return codes

Return code	Description
-2	No operation was performed.
-1	End user cancelled the operation.
0	Success
2	Error writing file
97	LiveEditor is required for this operation.

A.22 LiveSaveAs

The LiveSaveAs function saves the Live document with a different file name. You can use this function to ensure that end users do not save a revision that overwrites the original version of a Live document.

A.22.1 Syntax

`LiveSaveAs(DlfFileName)`

A.22.2 Arguments

LiveSaveAs argument

Argument	Data type	Use	Description
DlfFileName	String	Required	The file name under which the DLF file should be saved. The path and file name, if included, must appear in quotation marks.

A.22.3 Return Codes

LiveSaveAs returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveSaveAs return codes

Return code	Description
0	Success
2	Error writing file

A.23 LiveSaveAsPDF

The LiveSaveAsPDF function saves the Live document as a PDF. You can use arguments to provide specified save settings, such as the file name under which to save the document or a subset of pages to save. For example, if you add a "Save this document as PDF" button to the Live document, you can use the LiveSaveAsPDF function to specify the document you want to save. In this case, if you specified document 5 as the document to save in the function, only the fifth document in the file appears in the generated PDF.

Tip: When you are designing Live documents that contain imported PDFs and that end users will print locally, the `LiveSaveAsPDF` function can be used to obtain better print quality than is possible when you print directly from LiveEditor. In order to accommodate display on a monitor, LiveEditor converts imported PDFs into rasterized images and then displays those images on the screen. Likewise, LiveEditor uses rasterized images of imported PDFs when creating printed versions of Live documents. When creating PDF versions of Live documents that contain imported PDFs, however, LiveEditor uses the original imported PDF files. Therefore, if your Live document contains imported PDFs, the `LiveSaveAsPDF` function creates a PDF that yields better results for displaying, printing, and archiving than the original DLF file itself.

For more information about controlling how end users can save and print Live documents, see ["Controlling How End Users Save and Print Live Documents" on page 261](#).

Keep in mind that LiveEditor does not give end users the ability to specify ranges when they save a DLF file as a PDF. If you want to give them the ability to save specific pages, documents, or customers as a PDF, you must use the `LiveSaveAsPDF` function.

A.23.1 Syntax

`LiveSaveAsPDF(PDFFileName, Unit, Range)`

A.23.2 Arguments

LiveSaveAsPDF arguments

Argument	Data type	Use	Description
<code>PDFFileName</code>	String	Required	The file name under which the PDF should be saved. The path and file name, if included, must appear in quotation marks. It is recommended that you specify a file path and name; otherwise, the PDF is saved in the current working directory, which can change based on end user actions during the editing session.
<code>Unit</code>	String	Optional	The unit you want the Save as PDF dialog box to be set up to save. You can specify one of the following values: <ul style="list-style-type: none">• C or CUSTOMER• D or DOCUMENT• P or PAGE

LiveSaveAsPDF arguments, continued

Argument	Data type	Use	Description
Range	String	Optional	<p>The range of units to be saved (for example, you can allow end users to save documents 1 through 5). The range you specify for a page or document indicates the unit's location in the file (not in the current customer). For example, if you specify 5 as the range of the document to be saved, the fifth document in the file is saved, not the fifth document in the customer.</p> <p>You can specify ranges in any of the following ways:</p> <ul style="list-style-type: none">• Serial ranges—Use a dash to indicate the range (for example, 1-5). If you specify multiple overlapping ranges, only one copy of each unit is saved. For example, if you specify 1-4, 2-4, then the PDF contains pages 1-4 only.• Non-serial ranges—Use a comma to indicate the ranges (for example, 5, 10, 14).• A combination—Use dashes and commas to indicate the ranges (for example, 1-5, 10, 12-30). <p>If you do not specify a value, all the pages in the specified unit will be saved.</p>

For example, you can use the following function to save all of the pages for the first customer as a PDF:

```
LiveSaveAsPDF("C:\Program Files\PDFs\MyPDF.pdf", "C", "1")
```

You can use the following function to save pages 1–10 in the file:

```
LiveSaveAsPDF("C:\Program Files\PDFs\MyPDF.pdf", "P", "1-10")
```

A.23.3 Return Codes

LiveSaveAsPDF returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveSaveAs return codes

Return code	Description
0	Success
2	Error writing file

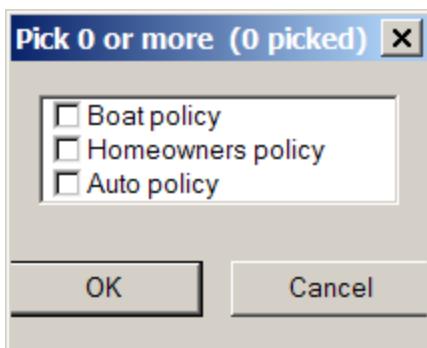
A.24 LiveSelection

The LiveSelection function opens a dialog box for the specified variable. End users can select a check box or radio button next to an area to include in the document.

Check boxes in the LiveEditor

- Buying boat
water and n
- Securing yo
knows that l
- Automobile
minimums :

Dialog box in the LiveEditor



The LiveSelection function returns a Boolean value of TRUE if the end user selects an option and clicks **OK**. Otherwise, it returns a value of FALSE.

A.24.1 Syntax

`LiveSelection(VariableName)`

A.24.2 Argument

LiveSelection argument

Argument	Data type	Use	Description
VariableName	Any	Required	Name of the variable. The variable cannot be a local variable.

A.24.3 Return Codes

LiveSelection returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveSelection return codes

Return code	Description
0	Success
97	LiveEditor is required for this operation.

A.25 LiveSendEmail

The LiveSendEmail function saves the current Live document or its XML content to a temporary file and sends the file as an email attachment. You can also send the XML content as the body of the email. The purpose of the LiveSendEmail function is to provide end users with an alternate method for submitting a Live document, which can be particularly useful for end users who are working outside of a company firewall.

You can use this function in one of three ways:

- You can send customer XML as part of the body of the email.
- You can send customer XML as an email attachment.
- You can send the DLF file as email attachment. A status other than 0 is returned only when the Interactive argument is FALSE. Messaging Application Programming Interface (MAPI) support is required on the editing computer.

Note: You can use LiveSendEmail only if MAPI is supported on the computer where LiveEditor is used.

A.25.1 Syntax

LiveSendEmail([Interactive, WhatToSend, Address, Subject, Line,...])

A.25.2 Arguments

LiveSendEmail arguments

Argument	Data type	Use	Description
Interactive	Boolean	Optional	The Boolean value specifying whether the document is interactive ("Live") after it is emailed. If the value is TRUE, then the document maintains its interactivity. If the value is FALSE, then the recipient cannot interact with it. The default value is TRUE.
WhatToSend	Integer	Optional	An integer used to specify what to send
Address	String	Optional	The email address where the DLF file should be sent (for example, <code>exstream@opentext.com</code>) To repeat the Address argument, separate multiple email addresses by a semi-colon (;). The Address argument cannot be blank if the value of Interactive is FALSE. The address must appear in quotation marks.
Subject	String	Optional	The subject line of the email. The text must appear in quotation marks.
Line	String	Optional	The body of the email. You can repeat the Line argument as many times as you need. The text must appear in quotation marks.

WhatToSend Argument

Specify the integer that corresponds with the item to be sent. The following table outlines the WhatToSend options available.

WhatToSend arguments

WhatToSend integer	Description
0	The current document is saved and sent as an email attachment.
1	A LiveXMLWrite function is performed and the document is sent as an email attachment.
2	The XML is sent in the body of the email.

A.25.3 Return Codes

LiveSendEmail returns an integer specifying the results of the requested operation. The following table lists the possible returns.

Note: A return code of 50, 51, or 52 is returned only when the value of the Interactive argument is FALSE.

LiveSendEmail return codes

Return code	Description
0	Success
50	Missing target address
51	Mail support unavailable
52	Send failed
97	LiveEditor is required for this operation.

A.26 LiveSetPrivacyMask

The `LiveSetPrivacyMask` function allows you to specify whether the privacy mask is enabled ("turned on") or disabled ("turned off"). The setting you specify (either enabled or disabled) can apply to all variables in the Live document, or you can specify a list of variables to which it applies.

You can also use `LiveSetPrivacyMask` in conjunction with user group information to reveal the actual value of a variable to a user in a specific user group. For example, you might want to allow an end user who belongs to the "Manager" or "Admin" group to see sensitive data, such as a Social Security number.

For more information about using privacy masks, see ["Protecting Variable Data by Using Privacy Masks" on page 308](#).

A.26.1 Syntax

`LiveSetPrivacyMask(OnOff, Variable,...,Variable)`

A.26.2 Arguments

LiveSetPrivacyMask arguments

Argument	Data type	Use	Description
OnOff	Boolean	Required	The Boolean value specifying whether the privacy mask is enabled or disabled, either for all variables or for the variables specified by the <code>Variable</code> argument. If the value is TRUE, then the privacy mask is enabled for the variables. If the value is FALSE, then the privacy mask is disabled for the variables.

LiveSetPrivacyMask arguments, continued

Argument	Data type	Use	Description
Variable	Variable	Optional	The name of the variable whose privacy mask you want to turn on or off. You can include as many variables as arguments as needed. Each variable must be separated by a comma. If you do not specify a variable, then the behavior you specified with the OnOff argument applies to all variables in the Live document.

A.26.3 Return Codes

LiveSetPrivacyMask return codes

Return code	Description
0	Privacy masks were disabled for specified variables.
1	An invalid argument (for example, one or more of the arguments were not variables)
2	No privacy mask (one or more of the variables supplied as an argument did not have a privacy mask enabled)

A.27 LiveSetReadOnly

The `LiveSetReadOnly` function sets the current Live document to read-only or restores the prior read-only state so that end users cannot make edits.

A.27.1 Syntax

`LiveSetReadOnly(ReadOnly)`

A.27.2 Argument

LiveSetReadOnly argument

Argument	Data type	Use	Description
ReadOnly	Boolean	Optional	The Boolean value specifying whether the document is set as read-only. If the value is TRUE, then the document becomes read-only and you cannot interact with it. If the value is FALSE, you can interact with the document. The default is TRUE.

A.28 LiveSetStyleSheet

The `LiveSetStyleSheet` function changes the current style sheet and reformats the open DLF file with the new styles. For example, suppose you work for a travel agency and are designing a brochure that details vacation destinations. Some content, such as the travel agent's name and phone number, is consistent no matter the destination. However, as end users (travel agents) select different destinations, you want each destination to have a distinctive appearance. In this scenario, you can use style sheets to make the brochure more visually appealing by associating the `LiveSetStyleSheet` function with each selection and changing the font to an ornate font for a "Royal Tour of England," or a Gothic font for a "Cathedrals of Paris" trip.

By default, when an end user changes style sheets, style changes occur at the page level. In other words, the end user must select a style sheet for each page in a Live document. If you want style sheets to apply style changes at the customer level and to all objects, you must select the **Styles are applied to all pages and objects** check box on the setting object associated with the Live document. When this check box is selected, styles are applied at the customer level whether you are using the menu option in the DLF or using the `LiveSetStyleSheet` function to set a style sheet.

For more information about style sheets, see ["Using Style Sheets to Provide Pre-Set Formatting Options" on page 258](#).

A.28.1 Syntax

`LiveSetStyleSheet(StyleSheetName)`

A.28.2 Argument

An argument is required when using the `LiveSetStyleSheet` function. Use one of the following arguments:

`LiveSetStyleSheet` argument

Argument	Data type	Use	Description
None	String	Optional	If you use the None argument, the DLF reverts to the style sheet that was applied when the DLF was first opened.
StyleSheetName	String	Optional	The name of the style sheet. The name of the style sheet must be in quotation marks.

A.28.3 Return Codes

`LiveSetStyleSheet` returns an integer specifying the results of the requested operation. The following table lists the possible returns.

`LiveSetStyleSheet` return codes

Return code	Description
0	Success
1	Specified design object not found.
97	LiveEditor is required for this operation.

A.29 LiveSign

The `LiveSign` function loads a new electronic signature into a Live document. When you use this function, the DLF file is locked according to the scope specified by the signature.

For more information about signatures in Live documents, see ["Using Signature Buttons to Allow Certain End Users to Prevent Changes in Live Documents"](#) on page 311.

A.29.1 Syntax

`LiveSign(Signature ID[, "Signature Name", "Scope"])`

A.29.2 Arguments

`LiveSign` arguments

Argument	Data type	Use	Description
Signature ID	Integer	Required	<p>The ID of the electronic signature. This value is usually the value of 'SYSLD_SignatureID'.</p> <p>Specify 0 to add a new electronic signature to the entire file.</p> <p>Tip: You can specify a wildcard for the <code>Signature Name</code> or <code>Scope</code> argument to control which signatures are signed based on the other argument.</p>

LiveSign arguments, continued

Argument	Data type	Use	Description
Signature Name	String	Optional	The name of the signature as specified in the Name box on the Advanced properties dialog box
Scope	String	Optional	The scope within the file that was locked by the signature. One of the following values is returned: <ul style="list-style-type: none">• File—Signature locked the entire file.• Customer—Signature locked the customer.• Document—Signature locked the document.• Page—Signature locked the page.• None—Signature did not lock anything.

A.30 LiveStore

The **LiveStore** function lets end users record changes to a document in an external repository. Because you can associate actions with events in Live (such as writing a report when a Live document is closed), you can use the **LiveStore** function to trigger those events when a DLF file is stored in an external repository.

The **LiveStore** function returns a Boolean value of TRUE if the store was successful. If you define multiple store events, then a value of TRUE is returned only if all stores are successful. **LiveStore** has no arguments. If you enter an argument, then you receive an error.

A.30.1 Syntax

`LiveStore()`

A.30.2 Return Codes

LiveStore returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveStore return codes

Return code	Description
0	Success
97	LiveEditor is required for this operation.

A.31 LiveSubmit

The `LiveSubmit` function lets end users submit their finished document.

The `LiveSubmit` function returns a Boolean value of TRUE if the submission was successful. If you define multiple submit events, a value of TRUE is returned only if all submissions are successful. `LiveSubmit` has no arguments. If you enter an argument, then you receive an error.

A.31.1 Syntax

`LiveSubmit()`

A.31.2 Return Codes

`LiveSubmit` returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveStore return codes

Return code	Description
0	Success
97	LiveEditor is required for this operation.

A.32 LiveTestMembership

The `LiveTestMembership` function checks whether the current LiveEditor user belongs to the specified design group. This function allows you to use design groups to control which objects specific end users can access. `LiveTestMembership` uses the end user's login information to determine whether the end user belongs to the specified design group.

For more information about controlling the editing of Live documents, see “[Controlling Access to Live Documents and Areas within Live Documents](#)” on page 286.

A.32.1 Syntax

`LiveTestMembership(Design group)`

A.32.2 Return Codes

`LiveTestMembership` returns an integer specifying the results of the requested operation. The following table lists the possible returns.

`LiveTestMembership` return codes

Return code	Description
0	Success. (The end user is a member of the design group.)
50	The end user is not in design group. (The user is not a member of the specified design group.)
51	The design group is unknown. (This situation might occur if the design group name is misspelled, or the group name was not added to the Group access box on the Authorization tab of the setting object, or the design group is not in one of the groups added to the Group access box.)
52	Live authorization is not set up. (This situation might occur if the setting object associated with the Live document did not define the authentication method.)

A.32.3 Return Codes with StatusString

If you use the `StatusString` built-in function to return a description of the return code (for example, to track them in a log file), the function returns the following text strings for each return code:

`StatusString` return codes

Return code	String
0	Operation successful
50	The Live user is not in the design group.
51	The design group is unknown.
52	Live authorization is not set up.

For more information about the `StatusString` built-in function, see *Using Logic to Drive an Application* in the Exstream Design and Production documentation.

A.33 LiveValidate

The `LiveValidate` function uses Live theme settings to highlight errors in logic and syntax.

The `LiveValidate` function returns a Boolean value of TRUE if the current Live document is valid (has no edit errors). Otherwise, it returns a value of FALSE.

For more information about Live themes, see “[Using Themes to Visually Guide End Users](#)” on [page 218](#).

A.33.1 Syntax

`LiveValidate(EntireFile)`

A.33.2 Argument

LiveValidate argument

Argument	Data type	Use	Description
EntireFile	Boolean	Optional	If the value is TRUE, then the entire file is validated. If the value is FALSE, then only the current customer is validated. The default is FALSE.

A.34 LiveView

The `LiveView` function changes the view used in LiveEditor. You might use this function if end user changes to a Live document require them to have access to a specific toolbar.

For more information about views, see “[Using Views to Customize the End-User Interface](#)” on [page 226](#).

A.34.1 Syntax

`LiveView(ViewName)`

A.34.2 Argument

LiveView argument

Argument	Data type	Use	Description
ViewName	String	Required	The name of the new view to be used. The name must be in quotation marks.

Value Arguments

The following table outlines the values that might be returned.

Value arguments

Value (Boolean)	Description
TRUE	The view loaded successfully.
FALSE	The view did not load.

A.35 LiveXMLRead

The `LiveXMLRead` function reads data from the specified XML file and sets variable values.

A.35.1 Syntax

```
LiveXMLRead(XmlFileName[, ShowMessages])
```

A.35.2 Arguments

LiveXMLRead arguments

Argument	Data type	Use	Description
XmlFileName	String	Required	The name of the XML file to be read. The name must be in quotation marks.
ShowMessages	Boolean	Optional	If the value of <code>ShowMessages</code> is TRUE, then the error and warning messages are shown. By default, the value is FALSE.

A.35.3 Return Codes

`LiveXMLRead` returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveXMLRead return codes

Return code	Description
0	Success
30	The specified file cannot be found.
31	The specified file is already open.
97	LiveEditor is required for this operation.

A.36 LiveXMLWrite

The `LiveXMLWrite` function writes variable values to the specified XML file.

A.36.1 Syntax

```
LiveXMLWrite(XmlFileName[, ShowMessages])
```

A.36.2 Arguments

LiveXMLWrite arguments

Argument	Data type	Use	Description
XmlFileName	String	Required	The name of the XML file where the variable values are written. The name must be in quotation marks.
ShowMessages	Boolean	Optional	The Boolean value. If the value of ShowMessages is TRUE, then the error and warning messages are shown. By default, the value is FALSE.

A.36.3 Return Codes

LiveXMLWrite returns an integer specifying the results of the requested operation. The following table lists the possible returns.

LiveXMLWrite return codes

Return code	Description
0	Success
2	Error writing file
97	LiveEditor is required for this operation.

Appendix B: Live System Variables

Like the system variables available as part of the Exstream Design and Production environment, system variables available in Exstream Live provide you with information that you can use to execute other logic. For example, system variables might provide user information that you can use to authenticate users, or file information that identifies the path of the Live document. All system variables are stored under the **Data Dictionary** heading under the Exstream Design and Production root folder.

This appendix describes the Live-specific system variables you can use in Exstream Live documents. The Exstream Design and Production system variables can also be used in Live documents; however, some Exstream Design and Production system variables have unique behaviors that you should be aware of before using them. These behaviors are also discussed in this appendix.

This chapter discusses the following topics:

- “[Comprehensive List of Live System Variables](#)” below
- “[Unique Behaviors of Date System Variables in LiveEditor](#)” on page 428
- “[Unique Behaviors of Page Numbering System Variables in Live Documents](#)” on page 429

B.1 Comprehensive List of Live System Variables

The following table lists all of the Live system variables available to you.

System variables for Live documents

System variable	Description
'SYSLD_ActiveControl'	The name of the active control (as entered on the Advanced properties dialog box). Use this variable only within a function assigned to a button action. You can use the 'SYSLD_ActiveControl' variable to create more generic buttons. For example, if you are using electronic signatures, you can create a single button that has functionality based on the electronic signature.
'SYSLD_ApplicationName'	The name of the application object that produced the Live document
'SYSLD_Content'	The content of the Live document
'SYSLD_ContentXML'	The content XML from the Live document
'SYSLD_CreatedBy'	The company or agency that originally created the Live document

System variables for Live documents, continued

System variable	Description
'SYSLD_CreationDateTime'	The date and time when the Live document was originally created. Format is SOAP timestamp.
'SYSLD_CurrentStyleSheet'	The style sheet currently in use in the Live document
'SYSLD_CustomerInFile'	The sequence number of the current customer in the Live document
'SYSLD_CustomersInFile'	The total number of customers included in the Live document
'SYSLD_DocumentInCustomer'	The sequence number of the current document in the current customer
'SYSLD_DocumentInFile'	The sequence number of the current document in the Live document
'SYSLD_DocumentsInCustomer'	The total number of documents included in the current customer
'SYSLD_DocumentsInFile'	The total number of documents included in the Live document
'SYSLD_DocumentType'	The type of Live document: <ul style="list-style-type: none"> • 0—Normal • 1—Template
'SYSLD_EngineVersion'	The version of the engine used to produce the Live document
'SYSLD_FileName'	The file name of the Live document
'SYSLD_FileNameWithPath'	The full file path name of the Live document
'SYSLD_LiveDocumentVersion'	The version of LiveEditor for which the Live document was created
'SYSLD_OriginalStyleSheet'	The style sheet in use when the Live document was created
'SYSLD_OutputName'	The name of the output object used to produce the Live document
'SYSLD_PackageFile'	The name of the package file used to produce the Live document
'SYSLD_PackageVersion'	The serialization version of the package file used to produce the Live document
'SYSLD_PageInCustomer'	The sequence number of the current page in the current customer
'SYSLD_PageInDocument'	The sequence number of the current page in the current document
'SYSLD_PageInFile'	The sequence number of the current page in the Live document
'SYSLD_PagesInCustomer'	The total number of pages included in the current customer
'SYSLD_PagesInDocument'	The total number of pages included in the current document

System variables for Live documents, continued

System variable	Description
'SYSLD_PagesInFile'	The total number of pages included in the Live document
'SYSLD_Rev_By'	The end user who made the revision
'SYSLD_Rev_DateTime'	The date and time of the revision. Format is SOAP timestamp.
'SYSLD_Rev_LiveVersion'	The version of the Live document when it was saved (each version is saved as a string in an array)
'SYSLD_Rev_Note'	A note entered by an end user who made the revision
'SYSLD_Rev_Number'	The number of a specific revision
'SYSLD_RevisionNumber'	The number of times the Live document has been saved, starting at 1 for its original creation
'SYSLD_SignatureID'	The ID of the signature associated with the active signature button
'SYSLD_SystemEvent'	The number of the most recent Live system event
'SYSLD_SystemEventDescription'	A description of the most recent Live system event
'SYSLD_UserGroups'	The user groups to which the end user belongs (each user group is saved as a string in an array). If authentication is not set up on the Live document, the array will be empty.
'SYSLD_UserName'	The end user's login name. If an end user logs in using a login dialog box, this value is the name the end user uses to log in. If the end user does not use a login dialog box, this value is the user name derived from the end user's system.

B.2 Unique Behaviors of Date System Variables in LiveEditor

Two Exstream date system variables, 'SYS_DateCurrent' and 'SYS_DateAsOf', have special functionality in LiveEditor. By default, these variables use the current date. End users can override the date during an edit session, but the next time an end user opens the Live document, the current date is substituted.

B.3 Unique Behaviors of Page Numbering System Variables in Live Documents

You use Exstream Design and Production system variables to add page numbers to a Live document, just as you do with non-Live designs. When used in a Live document, Exstream Design and Production system variables allow page numbers to be updated based on changes that occur in the document during editing (such as the addition of pages). In addition, if the Live document is used in fulfillment, the page numbers can be updated based on changes to the Live document (for example, if documents are added before the Live document in the final output, if the final output uses duplex settings, or if the document properties require page numbering changes). If page numbers will change during a production process, it might be helpful to inform end users that the page numbers they see during editing will be different from the page numbers customers will see.

Note: Page numbers and page number counts do not reflect pages that are hidden in a Live document.

In order to allow page numbering system variables to be updated throughout the Live document workflow, you must select **Initial Engine, Interactive Editor, and Final Engine** from the **Default variable substitution & rule execution time** drop-down list on the **Basic** tab of the setting properties. This option ensures that the variable values are updated and rules are executed during the first engine run, during editing in the LiveEditor, and during the last engine run.

Because page numbering system variables are often used to produce print output as part of complex production processes, there are many page numbering variables available to you for use. Sometimes, several of the variables can be used to achieve the same result in LiveEditor. On the other hand, the proper use of some variables depends on your Live document workflow. For example, Live documents are always treated as simplex documents in LiveEditor; therefore, the page and sheet values are the same when the file is being edited and viewed in LiveEditor. However, if the Live document is sent through a fulfillment process to a duplex printer, the page and sheet values can be different. Therefore, you should consider your final production process as you decide which page numbering variables to use.

The following table describes how page numbering variables behave in Live documents and can help you determine the appropriate variable to achieve the numbering you need.

Page numbering variables behavior in Live documents

Variable	Can restart?	Description
'SYS_PageInDocument' 'SYS_PagePhysicalInDocument' 'SYS_SheetInDocument'	No	The page number in the current customer
'SYS_PageTotalInDocument' 'SYS_PageTotalPhysicalInDocument' 'SYS_SheetTotalInDocument'	No	The total number of pages in the current customer
'SYS_PageInBreak' 'SYS_PageInQueue' 'SYS_SheetInBreak' 'SYS_SheetInQueue'	No	The page number in the Live document
'SYS_PagePrintedValue'	Yes	The page number to produce on the current page. The page number might be different from the page count if the page count restarts at the beginning of a document or if the document includes an interview page. Interview pages have a value of 0 when you use this variable.
'SYS_PageTotalInRun' 'SYS_SheetTotalInRun'	No	The total number of pages in the Live document
'SYS_PageTotalPrinted'	Yes	The last page number in the current customer. The page number might be different from 'SYS_PageTotalPhysicalInDocument' if the page count restarts at the beginning of a document or if the document includes an interview page. Interview pages have a value of 0 when you use this variable.
'SYS_DocPrintedValue'	Yes	The document number to produce on the current page. The number increases for each document included for a given customer. If the first document in a customer is an interview document or contains only pages that are hidden, the second document is numbered as 1. If you specify a prefix value in the Table of contents area on the Composition tab of the document properties, the value of 'SYS_DocPrintedValue' contains only the document number. <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9;"> Tip: You can use the document properties to instruct this value to reset to 1 for a specific document (for example, if you do not want the first document in a customer to count as document 1). </div>
'SYS_SubDocInDocument'	No	The number of the document in the current customer
'SYS_CustomerDocuments' 'SYS_TotalSubDocsInDocument'	No	The total number of documents for the current customer

Page numbering variables behavior in Live documents, continued

Variable	Can restart?	Description
'SYS_DocumentTotalInRun' 'SYS_SubDocumentInBreak' 'SYS_SubDocumentInQueue'	No	The number of documents in the Live document
'SYS_DocumentInBreak' 'SYS_DocumentInQueue' 'SYS_CustomerInRun'	No	The number of the current customer

For more information about restarting page numbering in designs, see *Designing Customer Communications* in the Exstream Design and Production documentation.

Appendix C: LiveEditor and Live Engine Switches

This chapter discusses engine switches specific to Exstream Live. Like the engine switches available as part of the Exstream Design and Production environment (which can also be used with Live), the Live-specific engine switches allow you to customize and manipulate Live applications.

This appendix discusses the following topics:

- “[LiveEditor Switches](#)” below
- “[Live Processing Switches](#)” on page 438
- “[Live Encryption Switches](#)” on page 304“[Live Encryption Switches](#)” on page 442

C.1 LiveEditor Switches

LiveEditor switches are instructions that you can use to control LiveEditor settings. You can use LiveEditor switches from the command prompt or in the control file for your application, or you can pass them as URL parameters when you open a DLF file in Microsoft Internet Explorer. For example, if you want to set the value of a variable, such as your name, you can use the INITVARSET switch to set the variable value to your name as soon as the Live document opens.

Note: To pass LiveEditor switches as URL parameters when you open a DLF file in Internet Explorer, use the following syntax:

url?LEswitch1&LEswitch2&LEswitch3

For example:

`http://temp.url/temp.dlf?INITVARSET=cur_session,123&INITVARSET=agent_num,876`

This section discusses the following switches that are used with LiveEditor:

- “[ACCEPT_ALL](#)” on the next page
- “[ACTION](#)” on page 434
- “[DDAFILEMAP](#)” on page 434
- “[DLFUSER and DLFPASSWORD](#)” on page 435

- “[DSNMAP](#)” on page 435
- “[FILEMAP](#)” on page 436
- “[INITVARSET](#)” on page 436
- “[PRODUCTLEVEL](#)” on page 437
- “[WEBSERVICEMAP](#)” on page 437

Caution: Do not use spaces between options, values, and successive file names in a command line, as this can cause errors at run time. For example, use -VERBOSE=ON instead of -VERBOSE = ON.

C.1.1 ACCEPT_ALL

Use the ACCEPT_ALL switch to automatically accept all of the revisions in a Live document without opening LiveEditor. For example, if you routinely use revision tracking to monitor changes in Live documents, but only certain documents require final review before you submit them for fulfillment, you can use this switch to accept all revisions and finalize any changes in the documents that do not need further review.

For more information about using revision tracking, see “[Using Revision Tracking to Monitor Changes Made to a Live Document](#)” on page 269.

When you use the ACCEPT_ALL switch, LiveEditor performs the operation in the background and does not open the document for editing. Each accepted revision is recorded in the revision history, along with the user name that is used to accept all revisions. If you provide a user name using the DLFUSER switch, then that user name appears in the revision history. If you do not use the DLFUSER switch, then the system user name appears in the revision history.

For more information about the DLFUSER switch, see “[DLFUSER and DLFPASSWORD](#)” on page 435.

If the Live document requires end users to enter revision comments, LiveEditor automatically adds a comment to the document, recording the user name (from the DLFUSER switch or the system) that is used to accept all revisions in the document and the date and the time at which you accepted all revisions.

You can view and record the result of the operation and any resulting errors by redirecting the error output stream at the command line.

This switch requires no argument.

For example:

```
Live.exe -ACCEPT_ALL "C:\Live Documents\MyLiveDocument.dlf" 2>AcceptAllLog.txt
```

C.1.2 ACTION

Use the ACTION switch to apply a specified action immediately when opening a Live document. For example, an action might be used to create an email message in your email program that you can use to send an edited Live document to a specific person.

The name of the action is the only required argument for the ACTION switch. If the name has spaces, enclose the name in quotation marks. For example:

```
-ACTION="Send e-mail"
```

Note: The action must be available in the Live document; otherwise LiveEditor ignores the switch.

For information about how to add an action to a Live document, see [“Using Actions to Automate Editing Tasks” on page 129](#).

Caution: The ACTION switch is ignored in LiveViewer.

C.1.3 DDAFILEMAP

Use the DDAFILEMAP switch to change the data file target to a dynamic data access (DDA) routine. When the file opens, LiveEditor checks for mapping, overrides it, and opens the correct file.

Use the following arguments with the DDAFILEMAP switch:

DDAFILEMAP arguments

Argument name	Use	Description
FileName	Required	The name of the data file
ProgramType	Required	The language used to create the module. Use one of the following options: <ul style="list-style-type: none">• ASM• COBOL• DLL (for C and C++)• PL1
Module	Required	The location of the DLL file
Routine	Required	The name of the DDA routine
BufferSize	Required	The size of the buffer (in bytes)
OpenParams	Optional	Parameters specific to the module

For example:

```
-DDAFILEMAP=MyDriverFile.dat,COBOL,  
C:\Program Files\OpenText\Exstream\MyDLL.dll,  
MyRoutine,5000,MyParams
```

C.1.4 DLFUSER and DLFPASSWORD

If you are required to log in to LiveEditor by means of a login screen, you can use the DLFUSER and DLFPASSWORD switches to supply your login information and bypass the login screen. If your credentials are invalid, LiveEditor prompts you to re-enter your credentials by means of the login screen.

Your login user name is the only argument for the DLFUSER switch, and your password is the only argument for the DLFPASSWORD switch. For example:

```
-DLFUSER=UserName  
-DLFPASSWORD=Password
```

Caution: You must use both DLFUSER and DLFPASSWORD. If either switch is missing, LiveEditor ignores the one you supply.

C.1.5 DSNMAP

Use the DSNMAP switch to change the data source when loading LiveEditor. When an end user selects **Live > Data file > Read > [Data file name]** from the LiveEditor Menu bar, LiveEditor checks for mapping in the default file, overrides it, and then opens the file you specified.

Use the following arguments with the DSNMAP switch:

DSNMAP arguments

Argument name	Use	Description
DSN	Required	The original data source name (DSN) of the data source
NewDSN	Required	The new DSN of the data source
UserName	Optional	The user name, if required, to log into the data source
Password	Optional	The password, if required, to log into the data source
Schema	Optional	The schema of the data source

For example:

```
-DSNMAP=OriginalDSN,NewDSN
```

C.1.6 FILEMAP

Use the FILEMAP switch to specify a different location for a reference file, initialization file, or report file that is available for use in a DLF file. For example, if you have an updated version of a reference file, you can use the FILEMAP switch to direct LiveEditor to use the updated file. Before using the FILEMAP switch on a data file, however, you must make the file available in the Live document by adding it as an authorized data file on the Live setting object.

For information about using authorized data files, see [“Adding Authorized Data Files to a Live Application to Allow End Users to Access Data” on page 153](#).

Use the following arguments with the FILEMAP switch:

FILEMAP arguments

Argument name	Use	Description
DataFileName	Required	The name of the data file
NewFileLocation	Required	The new location of the data file

For example:

```
-FILEMAP=MyData.dat,  
C:\Program Files (x86)\OpenText\Exstream\MyData.dat
```

When using the FILEMAP switch, keep in mind the following behaviors:

- The FILEMAP switch cannot be used to change the location of driver files.
- Any updates that result from applying the FILEMAP switch to an initialization file will be visible only in the Debugger, not in the Live document itself. The updated data does not appear in the Live document because the content from the initialization file is already composed prior to the FILEMAP switch being applied.
- Similarly, if you have selected the **Initial open** check box in the **IO times** area on the **Interactive** tab of the properties of a reference file or a report file, any changes that result from applying the FILEMAP switch will not appear in the Live document the first time it is opened after the switch is applied. The updated data does not appear initially because the content from the reference file or report file is already composed prior to the FILEMAP switch being applied. After the Live document is saved and reopened, however, the updated data will appear in the Live document.

C.1.7 INITVARSET

Use the INITVARSET switch to set specified customer or system variables to a specific value before a Live document is opened in LiveEditor. You can use this switch multiple times to set the value of multiple variables.

Use the following arguments with the INITVARSET switch:

INITVARSET arguments

Argument name	Use	Description
VariableName	Required	The name of the variable you want to set
Value	Required	The value of the variable

For example:

```
-INITVARSET=AccountNumber,00001  
-INITVARSET=CompanyState,KY
```

Caution: The INITVARSET switch is ignored in LiveViewer.

C.1.8 PRODUCTLEVEL

Use the PRODUCTLEVEL switch to specify the product capabilities of LiveEditor.

VIEWER is the only valid argument for the PRODUCTLEVEL switch. It opens a Live document at the LiveViewer level, so you can see how your design appears to LiveViewer users.

For example:

```
-PRODUCTLEVEL=VIEWER
```

Caution: If you use the PRODUCTLEVEL switch, then any switches that alter a DLF file (such as ACTION or INITVARSET) will not have an effect. This is because LiveViewer opens Live documents in read-only mode.

C.1.9 WEBSERVICEMAP

Use the WEBSERVICEMAP switch to change the URL, SOAPAction header, and type of file transfer to use for a data file using the web service (SOAP) format, or change the URL to use for a data file using the web service (RESTful) format. The switch is not supported on the z/OS platform.

For information about web service data files, see “[Using a Web Service to Exchange Data Between Live Documents and Other Enterprise Systems](#)” on page 188.

Use the following arguments with the WEBSERVICEMAP switch:

WEBSERVICEMAP arguments

Argument name	Use	Description
OldURL	Required	The original URL of the web service

WEBSERVICEMAP arguments, continued

Argument name	Use	Description
NewURL	Required	The new URL of the web service
NewSoapAction	Optional	<p>The new SOAPAction HTTP header</p> <p>This argument is valid only for web service (SOAP) data files.</p>
NewOptionsBitmap	Optional	<p>The new file transfer method. Use one of the following options:</p> <ul style="list-style-type: none">• 0—None• 1—Secure transaction (https)• 2—SOAP 1.1 specification (default)• 4—SOAP 1.2 specification <p>This argument is valid only for web service (SOAP) data files.</p>
CurrentSoapAction	Optional	<p>The current SOAPAction header. If this argument is specified, Live will attempt to perform the mapping using both the URL and the SOAPAction header.</p> <p>This argument is valid only for web service (SOAP) data files.</p>

For example:

```
-WEBSERVICEMAP=http://oldRESTurl.com,http://newRESTurl.com
-WEBSERVICEMAP=http://oldSOAPurl.com,http://newSOAPurl.com,
  http://newSOAPurl.com/NewSOAPAction,,,
  http://oldSOAPurl.com/OldSOAPAction
```

C.2 Live Processing Switches

The switches discussed in this section can be used from the command prompt or from a control file during engine processing. While you can use many of the Exstream Design and Production engine switches when you create Live documents, the switches discussed here are specific to the Live environment.

This section discusses the following Live switches that are used during processing:

- “[DISABLE_EMBED_DATA](#)” on the next page
- “[DISABLE_FLOW_TARGETS_FOR_LIVE](#)” on the next page
- “[DLFCAMPDISABLE](#)” on the next page
- “[DRIVERLISTFILE](#)” on page 440
- “[PROMOTECAMPTODLF](#)” on page 440
- “[RETAIN_VARIABLE_RESET_TIME](#)” on page 441

- “[SHOWREVISION](#)” on page 441
- “[USE_DLF_STYLE](#)” on page 442

C.2.1 DISABLE_EMBED_DATA

Use the DISABLE_EMBED_DATA engine switch to disable the use of an embedded reference file in a fulfillment application. The only argument the DISABLE_EMBED_DATA engine switch supports is the object OI for the embedded reference file it is disabling. For example, to disable the use of a reference file with the Internal ID of 3, use the syntax -DISABLE_EMBED_DATA=3.

Tip: To find the object OI, right-click on the reference file and select **Administer**. The number in the **Internal ID** field is the OI for the object.

For information about embedding data files in a Live document, see “[Embedding Data Files](#)” on [page 151](#).

C.2.2 DISABLE_FLOW_TARGETS_FOR_LIVE

Use the DISABLE_FLOW_TARGETS_FOR_LIVE switch if you have designated flow frame targets for an application, but you want the engine to ignore the **Enable flow frame targeting** check box when producing DLF output. If you use this switch, keep in mind that the page count in the engine message file might be inaccurate.

This switch affects only DLF output.

For more information about designating content to flow to a specific frame, see *Designing Customer Communications* in the Exstream Design and Production documentation.

For more information about using flow frames in Live documents (DLF output), see *Designing for LiveEditor* in the Exstream Design and Production documentation.

This switch requires no arguments.

For example:

-DISABLE_FLOW_TARGETS_FOR_LIVE

C.2.3 DLFCAMPDISABLE

Use the DLFCAMPDISABLE switch to suppress a campaign that has been added to a Live document. You can use this switch in combination with the PROMOTECAMPTODLF switch to replace outdated campaigns. For example, suppose an end user places messages from Campaign A in a Live document, but after you distributed the Live document to end users, Campaign A was dropped in favor of Campaign B. Instead of having to redistribute the Live document and have end users replace their campaign selections, you can first use the

DLFCAMPDISABLE switch to suppress Campaign A, and then use the PROMOTECAMPTODLF switch to place messages from Campaign B.

Keep in mind that you must also use the DLFCAMPDISABLE and PROMOTECAMPTODLF switches to update existing campaigns (for example, to replace an outdated version of Campaign A with an updated version of the same campaign).

The syntax of the switch must be structured as follows:

```
-DLFCAMPDISABLE=<campaign name>
```

For information about using campaigns and messages in Live, see [“Managing Messaging and Marketing Content in Live Documents” on page 208](#).

C.2.4 DRIVERLISTFILE

Use the DRIVERLISTFILE switch to identify the location of a text file that identifies a list of driver files. The file specified by the DRIVERLISTFILE switch must contain the absolute or relative path to each Live document to be used as a driver file (with one path on each line). In order to be processed in a single batch run, all the Live documents must have the same layout. Therefore, if the engine encounters a Live document that does not match the layout specified in the fulfillment application, you receive an error. When the Live documents are processed, they inherit the data file properties of the primary driver file that provides the XML layout in the fulfillment application.

Sample file containing list of Live documents

```
C:\Program Files\Exstream\CustomerWelcome003423.DLF
C:\Program Files\Exstream\CustomerWelcome003553.DLF
C:\Program Files\Exstream\CustomerWelcome003801.DLF
C:\Program Files\Exstream\CustomerWelcome004463.DLF
C:\Program Files\Exstream\CustomerWelcome004493.DLF
C:\Program Files\Exstream\CustomerWelcome004528.DLF
C:\Program Files\Exstream\CustomerWelcome004687.DLF
C:\Program Files\Exstream\CustomerWelcome005923.DLF
C:\Program Files\Exstream\CustomerWelcome006702.DLF
C:\Program Files\Exstream\CustomerWelcome006723.DLF
```

The syntax of the switch must be structured as follows:

```
-DRIVERLISTFILE=C:\Program Files\OpenText\DriverList.txt
```

For information about using multiple Live documents as driver files, see [“Using Multiple Live Documents as Driver Files” on page 363](#).

C.2.5 PROMOTECAMPTODLF

Use the PROMOTECAMPTODLF switch to replace a campaign that has been added to a Live document. You can use this switch in combination with the DLFCAMPDISABLE switch to

replace outdated campaigns.

The syntax of the switch must be structured as follows:

-PROMOTECAMPTODLF=<campaign name>

For information about the DLFCAMPDISABLE switch, see “[DLFCAMPDISABLE](#)” on page 439.

For information about using campaigns and messages in Live, see “[Managing Messaging and Marketing Content in Live Documents](#)” on page 208.

C.2.6 RETAIN_VARIABLE_RESET_TIME

Use the RETAIN_VARIABLE_RESET_TIME switch to ensure that the non-key variable reset time used in the engine run that generates a DLF file is retained in the fulfillment run. The RETAIN_VARIABLE_RESET_TIME switch overrides the default behavior, which is to ignore variable reset times in the fulfillment run.

This switch requires no arguments.

For example:

RETAIN_VARIABLE_RESET_TIME

C.2.7 SHOWREVISION

Use the SHOWREVISION switch to specify which version of a Live document to produce during fulfillment, when the Live document uses revision tracking and includes tracked changes. Using this switch lets you control whether unapproved changes appear in the final output after fulfillment.

For more information about using revision tracking, see “[Using Revision Tracking to Monitor Changes Made to a Live Document](#)” on page 269.

The argument for this switch is the version of the document that you want to produce. The default value is CURRENT. Use one of the following options:

- CURRENT—Produces the final document (as it would appear after all changes from all users were approved)
- ORIGINAL—Produces the original document (as it appeared before tracked changes were made)
- MARKUP—Produces a draft that includes both the original content and the changed content

For example:

-SHOWREVISION=ORIGINAL

C.2.8 USE_DLF_STYLE

Use the USE_DLF_STYLE engine switch to disable the use of the SYS_StyleSheet variable as a formula in a fulfillment application. If you are using a DLF that contains style sheets as a driver file in a fulfillment application, then you must use the USE_DLF_STYLE engine switch to prevent the fulfillment application from recalculating the SYS_StyleSheet value and overwriting the changes from the DLF. When you use this switch, the SYS_StyleSheet variable can still be mapped in a fulfillment application in a data file.

For more information about using style sheets in Live applications, see [“Using Style Sheets to Provide Pre-Set Formatting Options” on page 258](#).

C.3 Live Encryption Switches

The switches discussed in this section can be used from the command prompt or from a control file to encrypt or decrypt XML content within Live documents. The switches discussed here are specific to the Live environment, and they are used separately from any processing switches.

For information about encrypting Live documents, see [“Encrypting Data and Content in a Live Document” on page 303](#).

When using Live encryption switches, you must also use the KEY switch, KEYFILE switch, or KEYPART switch to specify a key.

For more information about using switches to specify a key, see *Preparing Applications for Production* in the Exstream Design and Production documentation.

This section discusses the following switches that are used to manage Live document encryption:

- [“DLF_CIPHER” below](#)
- [“DLF_CIPHERLIST” on the next page](#)
- [“DLF_CIPHERLOG” on page 444](#)
- [“DLF_CIPHEROPTIONS” on page 444](#)
- [“DLF_CIPHEROUT” on page 445](#)

C.3.1 DLF_CIPHER

Use the DLF_CIPHER switch to encrypt or decrypt a single DLF file.

Use the following arguments with the DLF_CIPHER switch:

Argument name	Use	Description
Mode	Required	This argument specifies whether the Live document should be encrypted or decrypted. Use one of the following options: <ul style="list-style-type: none">• ENCRYPT• DECRYPT
FileLocation	Required	The location of the DLF file

For example:

```
-DLF_CIPHER=DECRYPT,C:\DLFs\LiveDocument.dlf
```

C.3.2 DLF_CIPHERLIST

Use the DLF_CIPHERLIST switch to encrypt or decrypt multiple DLF files listed in a separate text file. The file specified by the DLF_CIPHERLIST switch must contain the absolute or relative path to each DLF file to be encrypted or decrypted (with one path on each line).

Sample file containing list of DLF files

```
C:\Program Files\Exstream\CustomerWelcome003423.DLF
C:\Program Files\Exstream\CustomerWelcome003553.DLF
C:\Program Files\Exstream\CustomerWelcome003801.DLF
C:\Program Files\Exstream\CustomerWelcome004463.DLF
C:\Program Files\Exstream\CustomerWelcome004493.DLF
C:\Program Files\Exstream\CustomerWelcome004528.DLF
C:\Program Files\Exstream\CustomerWelcome004687.DLF
C:\Program Files\Exstream\CustomerWelcome005923.DLF
C:\Program Files\Exstream\CustomerWelcome006702.DLF
C:\Program Files\Exstream\CustomerWelcome006723.DLF
```

Use the following arguments with the DLF_CIPHERLIST switch:

Argument name	Use	Description
Mode	Required	This argument specifies whether the Live documents should be encrypted or decrypted. Use one of the following options: <ul style="list-style-type: none">• ENCRYPT• DECRYPT
ListFileLocation	Required	The location of a text file that contains a list of DLF file locations

For example:

```
-DLF_CIPHERLIST=DECRYPT, C:\Program Files\OpenText\CustomerWelcomeList.txt
```

If you are working in a Multiple Virtual Storage (MVS) environment, you cannot use the DLF_CIPHEROUT switch to specify a different output location for multiple DLF files listed in a file specified by the DLF_CIPHERLIST switch, and the engine will overwrite the specified DLF files with the encrypted or decrypted versions.

C.3.3 DLF_CIPHERLOG

Use the DLF_CIPHERLOG switch to specify a location for the log of the encryption or decryption event. If you do not use this switch, the log file is written to the same folder as the engine.

The logged items are as follows:

- Engine version
- Date
- Mode (ENCRYPT or DECRYPT)
- Each file encrypted or decrypted and whether the action is completed successfully

The only argument is the log file location.

For example:

```
-DLF_CIPHERLOG=C:\logs\CipherLog.txt
```

C.3.4 DLF_CIPHEROPTIONS

Use the DLF_CIPHEROPTIONS switch to specify which data in the Live document should be encrypted or decrypted. If you do not use this switch, the action applies to all XML content within the Live document.

The only argument is the content to be processed. You can specify multiple options. Use one of the following options:

- CUSTOMER—The XML content related to the data stored for each customer
- CONTENTMANIFEST—The XML content related to the content stored within the Live document
- METADATA—The XML content related to the metadata about the Live document
- INTERVIEW—The XML content included on interview pages
- ALL—All XML content within the Live document

For example:

```
-DLF_CIPHEROPTIONS=CUSTOMER,METADATA
```

C.3.5 DLF_CIPHEROUT

Use the DLF_CIPHEROUT switch to specify a location for the encrypted or decrypted output file. If this switch is omitted, the engine overwrites the original file with the encrypted or decrypted version. Keep in mind that if you are processing multiple DLF files, only the file path specified is used, and any file name is ignored.

The only argument is the file location.

For example:

```
-DLF_CIPHEROUT=C:\DLFs\DecryptedFile.DLF
```

If you are working in a Multiple Virtual Storage (MVS) environment, you can use the DLF_CIPHEROUT switch only when you encrypt or decrypt a single Live document using the DLF_CIPHER switch. If you specify multiple Live documents using the DLF_CIPHERLIST switch, you cannot use the DLF_CIPHEROUT switch, and the engine will overwrite the specified DLF files with the encrypted or decrypted versions.