SeeBeyond ICAN Suite

# eVision Studio User's Guide

*Release 5.0.4*

SEEBEYOND®

Version 20040603192255.

# Contents

# List of Figures

# List of Tables

Chapter 1

# Introduction

*eVision Studio (eVision)* is a graphical design studio for the creation of integrated Web applications. eVision provides graphical abstractions of backend data, and modeling of user/system interactions. With eVision, Web developers can rapidly create Web applications that can be deployed standalone, or as a channel in a portal without requiring any special programming expertise.

eVision is a component of the SeeBeyond® Integrated Composite Application Network Suite™ (ICAN). The eVision component opens a real-time, interactive window into the ICAN Suite run-time environment. eVision allows the organization to present a single, unified view of enterprise data and applications to employees, customers, and partners.

A *Page Flow* is a series of Web pages that are laid out in a sequence to accomplish a specific group of tasks. User actions determine how the flow moves from page to page. Web applications enable the distribution of integrated Page Flows across the enterprise and allow real-time user interaction with those processes.

eVision Web applications receive and send data to ICAN Suite components, allowing users to interact with that data at run-time. Using eVision's rich set of Web development tools, the Web developer can create personalized views of business information, allowing Web application users to interact with running Page Flows in real time, while working with only the information that they need to see.

## 1.1 Document Purpose and Scope

The *eVision Studio User's Guide* explains how to use the eVision Studio application to create and deploy Web applications.

## 1.2 Organization of Information

The *eVision Studio User's Guide* includes the following information:

- A description of eVision Studio's user interface
- A description of eVision Studio's Page Layout Designer tools
- Instructions for creating sample Web pages
- Instructions for moving and resizing GUI components on the canvas
- A description of eVision Studio's Page Flow Designer tools
- Instructions for creating a Page Flow from pre-built components

## 1.3 Writing Conventions

The following writing conventions are observed throughout this document.

**Table 1** Writing Conventions

| Text | Convention | Example |
|---|---|---|
| Button, file, icon, parameter, variable, method, menu, and object names. | **Bold** text | ▪ Click **OK** to save and close.<br>▪ From the **File** menu, select **Exit**.<br>▪ Select the **logicalhost.exe** file.<br>▪ Enter the **timeout** value.<br>▪ Use the **getClassName()** method.<br>▪ Configure the **Inbound** File eWay. |
| Command line arguments and code samples | `Fixed` font. Variables are shown in ***`bold italic`***. | `bootstrap -p` ***`password`*** |
| Hypertext links | **Blue** text | For example: See **"Online Documentation" on page 18**. |

### Additional Conventions

#### Windows Systems

For the purposes of this guide, references to "Windows" will apply to Microsoft Windows Server 2003, Windows XP, and Windows 2000.

#### Path Name Separator

This guide uses the backslash ("\“) as the separator within path names. If you are working on a UNIX system, please make the appropriate substitutions.

## 1.4 Online Documentation

The documentation for the SeeBeyond ICAN Suite is distributed as a collection of online documents. These documents are viewable with the Acrobat Reader application from Adobe Systems. Acrobat Reader can be downloaded from:

**http://www.adobe.com**

## 1.5 The SeeBeyond Web Site

The SeeBeyond Web site is your best source for up-to-the-minute product news and technical support information. The site's URL is:

**http://www.seebeyond.com**

Chapter 2

# About eVision Studio

This chapter provides an overview of eVision Studio.

**This Chapter Includes:**

## 2.1 Summary of Features

eVision Studio is a graphical design studio that allows the business analyst and Web developer to create Web applications. With eVision, the Web developer can rapidly create interactive Web applications, which can be deployed standalone or as a channel within a portal, without requiring advanced programming abilities.

eVision's GUI components gather input data from the user at run-time, and add functions and interactivity to Web pages. GUI components (called HTML and Form Objects) comprise familiar Web-centric design elements, including check boxes, text boxes, graphics containers, horizontal (separator) lines, and submit buttons. GUI components are pre-built combinations of Java classes and JSP code that represent Web interface elements. Property sheets allow the developer to add the labeling, functionality, and appearance attributes that the components will display to users at run-time.

Through the use of familiar drag-and-drop techniques and property sheets, eVision allows the Web developer to interactively add graphics, text, and programmatic content to Web application pages. Pre-built components are dragged from convenient component palettes and positioned on the design canvas. Functional and appearance attributes are added to the components in the properties window. As a Web page is designed, the developer can preview it in a browser at any time throughout the process.

Using eVision's design tools and pre-built components, the Web application developer can create personalized views of business information, so that users can influence the flow of business information through the browser.

For example, eVision Web applications can be structured to allow employees to log into a Web server, see the tasks that are assigned to them, and then use the browser to complete their assignments. eVision opens a real-time, interactive window into the ICAN Suite run-time environment, presenting Web applications to users across the enterprise. eVision Web applications allow users to interact with business activities through the browser, and complete the business tasks that require real-time human analysis, judgement, and intervention.

## 2.2 eVision and the ICAN Suite

eVision is a component in the SeeBeyond ICAN Suite of applications. eVision provides Web application design and deployment features and functions to the ICAN Suite.

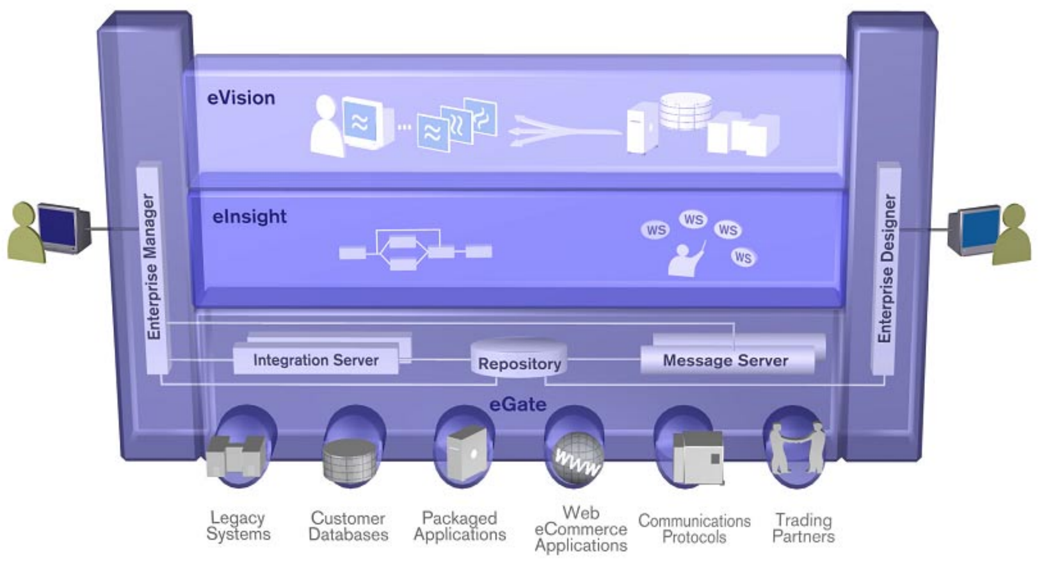
eVision simplifies the task of developing Web applications by using patterns and metaphors that are familiar to Web developers, providing graphical abstractions of back-end data, and modeling of user/system interactions.

### 2.2.1 ICAN Suite Integration

eVision is tightly integrated with the ICAN Suite and runs as a component within the ICAN Suite environment. The Page Layout Designer and Page Flow Designer run as components within the Enterprise Designer GUI. eVision Web applications are stored in the SeeBeyond Repository.

The following figure shows how eVision is integrated with the ICAN Suite.

**Figure 1** eVision Studio Integration With the ICAN Suite



- eVision Studio runs as a component within **Enterprise Designer**.
- Page Layouts, Page Flows, GUI components, and Deployment Profiles are stored in the **eGate Repository**.
- The Page Flow Engine, which coordinates all process-related activity of a deployed project, runs in a **SeeBeyond Integration Server**.

## 2.3 eVision Overview

eVision applications receive and send data to ICAN Suite components, allowing users to interact with that data at run-time. Web applications enable the distribution of integrated business processes across the enterprise and allow real-time user interaction with those processes. eVision allows the organization to present a single, unified view of enterprise data and applications to employees, customers, and partners.

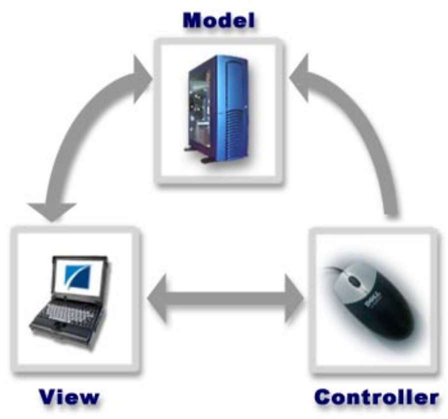
With eVision, the Web developer can create personalized views of business information, allowing Web application users to interact with running business processes in real time, while working with only the information that they need to see. Web applications can be easily structured to allow employees to see what tasks are assigned to them, then to interact with and complete the tasks.

## 2.4 MVC Architecture

eVision applications are based on the Model/View/Controller (MVC) architecture. MVC is a software development paradigm that enhances the task of building software systems, particularly those that generate multiple, synchronized presentations of the same data. For example, MVC is ideal for the development of a graphical statistical presentation application that requires simultaneous rendering of the same data in bar, line, and pie chart formats.

The MVC architecture consists of three types of objects: the Model, the View, and the Controller.

**Figure 2** Model/View/Controller (MVC) Architecture



The **Model** object represents the data in a program, which manages behaviors and data within an application. The Model responds to requests for information about its current state (typically requested by the View) and responds to instructions to change its state (typically requested by the Controller).

The **View** object manages the visual display of the Model data; for example, displaying graphics and text to users in a browser.

The **Controller** object enables user interaction with the Model data; for example, mouse and keyboard inputs from the user, which instruct the Model and/or View to perform an action.

eVision's application architecture fully supports the MVC paradigm. In an eVision Web application, user input, modeling of the external world, and visual feedback are managed by MVC objects, where each object is specialized for its task. For example:

- The **Model**, represented by the eVision **Page Flow**, contains the business logic (Object Type Definitions and Collaborations) that interacts with the back-end system applications.
- The **View** contains the JavaServer Pages (JSPs) that are generated with the **Page Layout Designer**.
- The **Controller** is the Web-enabled Page Flow created with the **Page Flow Designer**. The Controller orchestrates the sequence of pages being sent to the browser in response to user actions.

## 2.5 Process Overview

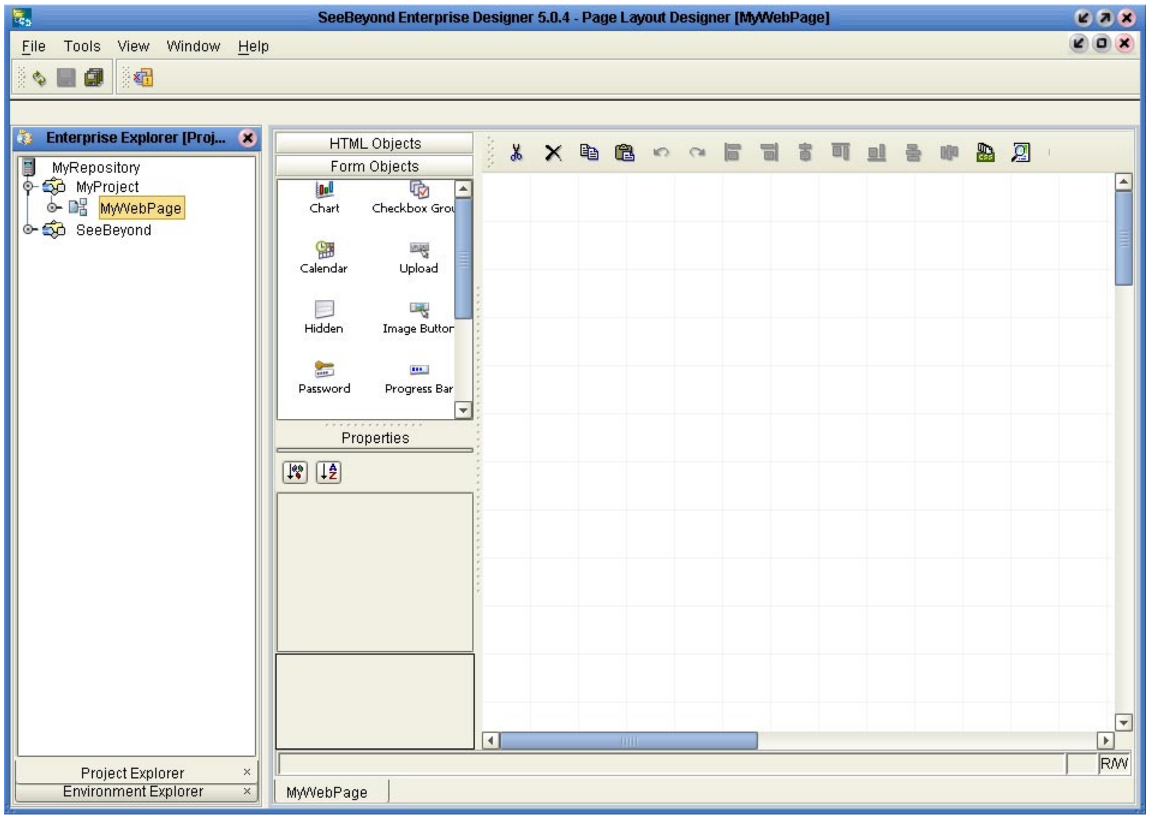
The steps for the Web application development process are:

- **Page Layout** - Using the **Page Layout Designer** to create Web pages with pre-built GUI components.
- **Page Linking** - Using the **Page Link Wizard** to create links to Web pages and pass parameters from one page to another.
- **Page Flow** - Using the **Page Flow Designer** to connect finished Web pages to create a logical flow for the underlying business process.
- **Binding and Deployment** - Deploying Web applications into an Environment for integration with other ICAN Suite run-time components.

### 2.5.1 User Interface Components

eVision Studio leverages the familiar Enterprise Designer interface to create, manage, deploy, and integrate Web applications.

**Figure 3** Page Layout Designer in Enterprise Designer



#### Page Layout Designer

The Page Layout Designer allows the Web designer to specify the details of individual pages in the application using familiar drag-and-drop techniques to place GUI components (Web page design objects) onto a design canvas. The Page Layout Designer provides a comprehensive collection of pre-built GUI components, and a friendly, WYSIWYG Web page editor.

#### Page Flow Designer

The Page Flow Designer facilitates the flow of Web-based business activities. The functions include Page Flow modeling, monitoring, and execution, as well as the ability to analyze how data messages flow from page to page.

The Page Flow Designer allows the business analyst to lay out the user workflow of a Web application by modeling the high-level, logical, page-by-page flow that users will follow through the Web application to complete a given task (the *Page Flow*). The business analyst can easily lay out a Web application's page flow using the Page Flow Designer's set of intuitive, graphical modeling tools. At run-time, the Page Flow drives the display of eVision Web pages and orchestrates the interactions with back-end systems.

Through the deployment of Web pages that are developed with the Page Layout Designer and Page Flow Designer tools, the business analyst can enable enterprise-wide, internal and external access to Web applications across an intranet, or the Internet.

### 2.5.2 Page Flow Engine

eVision Pages and Page Flows are Web application elements that allow users to interact with back-end systems to execute tasks that require human analysis and intervention. A typical example of human intervention in this context is credit approval based on a credit report. A Page Flow is typically engineered to guide the user through a page-by-page process of data viewing and task execution.

At run-time, Page Flows are evaluated and are then assigned to the proper group, user, or role. The assignee uses the eVision Web application (a Page or Page Flow) to finish the required tasks.

The **Page Flow Engine** orchestrates the system responses to the execution of Web page component code. At run-time, the Page Flow Engine executes page links, receives and processes user input, and, based on human interaction, moves the viewer from page to page until the underlying process is complete.

## 2.6 UTF-8 Support

eVision Studio provides support for the UTF-8 encoding scheme during both design time and run time. Therefore, an eVision Web application can include any character set that is supported by UTF-8, including Asian language character sets.

# Chapter 3
# Installing eVision Studio

This chapter describes how to install eVision Studio.

**This Chapter Includes:**

## 3.1 System Requirements

eVision Studio is available on the following operating systems:

- Windows Server 2003, Windows XP SP1a, and Windows 2000 SP3 and SP4
- HP Tru64 V5.1A
- HP-UX 11.0 and 11i (PA-RISC)
- IBM AIX 5.1L and 5.2
- Red Hat Linux Advanced Server 2.1 (Intel x86)
- Red Hat Linux 8 (Intel x86)
- Sun Solaris 8 and 9

The **Readme.txt** file on the CD-ROM contains the most up-to-date operating system requirements for the supported platforms.

For more information on platform support, see the *SeeBeyond ICAN Suite Installation Guide*.

### 3.1.1. Database Support

If desired, you can persist eVision Page Flows using a database. eVision supports the following databases:

- Oracle 8.1.7, 9.1, and 9.2
- Sybase 12.5
- Microsoft SQL Server 2000
- IBM DB2 Universal Database 8.1

## 3.2 Installation Steps

The installation procedure for eVision Studio is similar to the installation procedure for other ICAN Suite products. You can find general product installation instructions in the *SeeBeyond ICAN Suite Installation Guide*, which is available from Enterprise Manager's Documentation page.

Before you begin, ensure that the Repository is running.

### 3.2.1. Uploading eVision Studio to the Repository

eVision Studio consists of two **.sar** files:

- The **eVision.sar** file contains the product software.
- The **eVisionDocs.sar** file contains the documentation and sample files.

The following procedure must be performed by the **Administrator** user, or by a user that has been granted a role with upload privileges.

**To upload eVision Studio to the Repository**

1. Start Internet Explorer.
2. In the **Address** field, enter **http://*hostname*:*portnumber***

   where

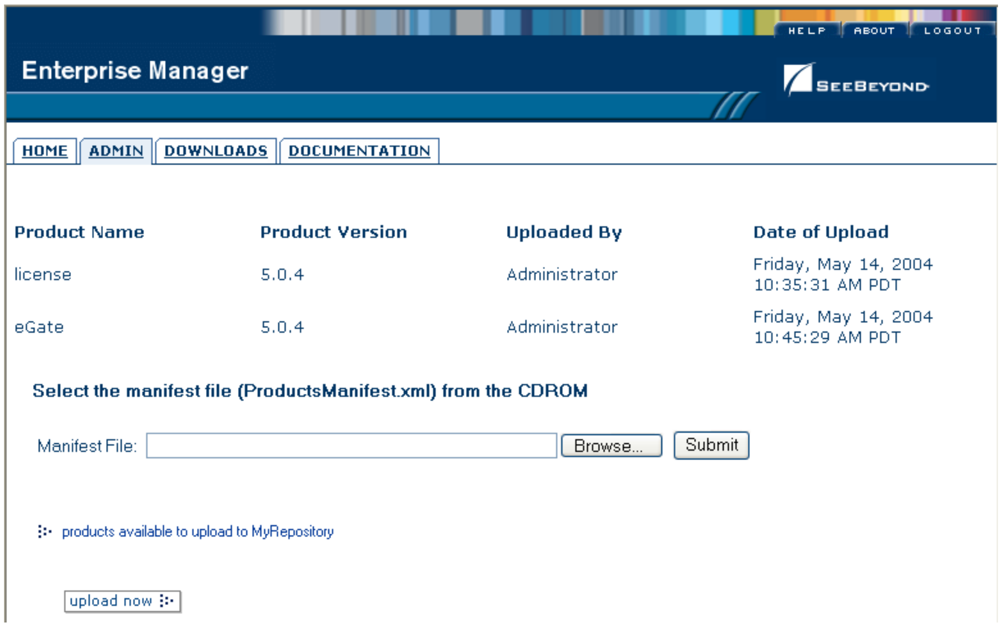
   ***hostname*** is the TCP/IP host name of the server where the Repository is installed.

   ***portnumber*** is the base port number of the Repository.

   The **SeeBeyond Customer Login** window of Enterprise Manager appears.
3. Enter your username and password.
4. Click **Login**.

   The Enterprise Manager home page appears.
5. Click the **ADMIN** tab. See Figure 4.

**Figure 4** Enterprise Manager ADMIN Page



| Product Name | Product Version | Uploaded By | Date of Upload |
|---|---|---|---|
| license | 5.0.4 | Administrator | Friday, May 14, 2004 10:35:31 AM PDT |
| eGate | 5.0.4 | Administrator | Friday, May 14, 2004 10:45:29 AM PDT |

Select the manifest file (ProductsManifest.xml) from the CDROM

Manifest File:

6 Click **Browse**. The **Choose file** dialog box appears.

7 Select the **ProductsManifest.xml** file in the top level of the Products - Disc 1 CD-ROM.

8 Click **Open**.

You are returned to the Enterprise Manager **ADMIN** page.

9 Click **Submit**.

The list of products available for uploading appears.

10 On the eVision row, click **Browse**. The **Choose file** dialog box appears.

11 Select the **eVision.sar** file.

12 Click **Open**.

You are returned to the Enterprise Manager **ADMIN** page.

13 Click **upload now**. See Figure 5.

**Figure 5** upload now Button

The progress bar indicates the status of the upload.

14 When the upload is finished, click **Back to component installation**.

15 To upload the eVision Studio documentation and sample files to the Repository, repeat steps 6 through 14 with the following differences:

- Use the **ProductsManifest.xml** file in the Documentation subdirectory.
- Upload the **eVisionDocs.sar** file in the Documentation subdirectory.

The documentation and sample files will be accessible from the **DOCUMENTATION** page.

## 3.2.2. Updating Enterprise Designer with eVision Modules

The following procedure adds the eVision modules to Enterprise Designer.

**To update Enterprise Designer with eVision modules**

1 On the client computer where Enterprise Designer is installed, start Enterprise Designer.

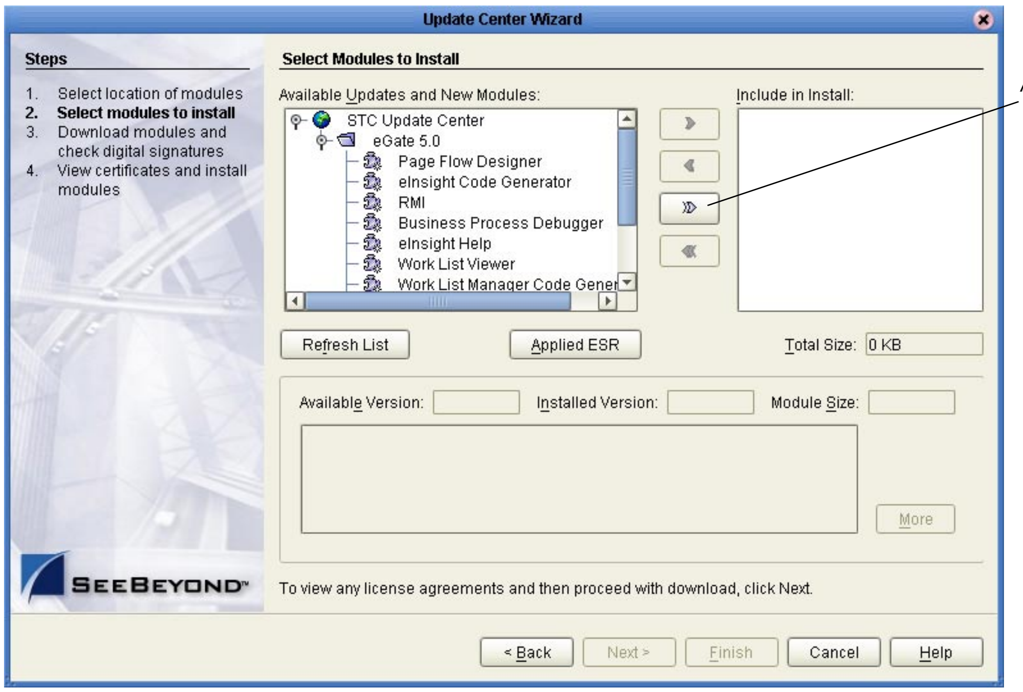
2 On the **Tools** menu, click **Update Center**.

The Update Center Wizard appears.

3 Click **Next**.

Step 2 appears. See Figure 6.

**Figure 6** Update Center Wizard: Select Modules to Install



To view any license agreements and then proceed with download, click Next.

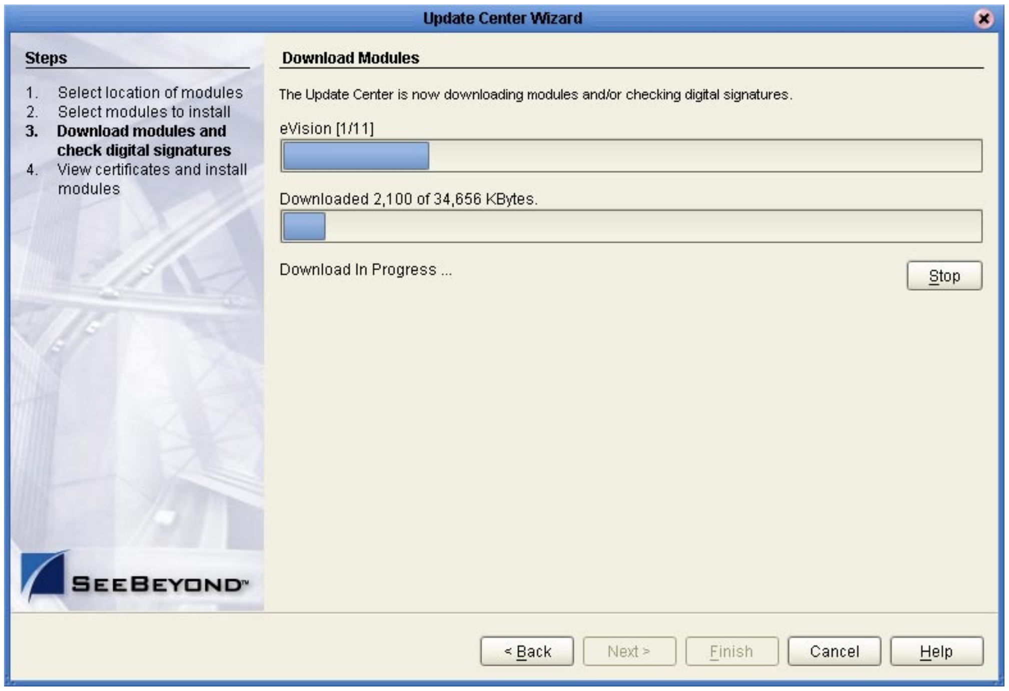
4 Click the **Add All** button. The eVision components move to the **Include in Install** box.

5 Click **Next**. The **License Agreement** window appears.

6 Click **Accept**.

The progress bars indicate the status of the download. See Figure 7.

**Figure 7** Update Center Wizard: Download Modules


Update Center Wizard
Steps
1. Select location of modules
2. Select modules to install
3. Download modules and check digital signatures
4. View certificates and install modules
SeeBeyond
Download Modules
Stop
< Back
Next >
Finish
Cancel
Help

The Update Center is now downloading modules and/or checking digital signatures.

eVision [1/11]

Downloaded 2,100 of 34,656 KBytes.

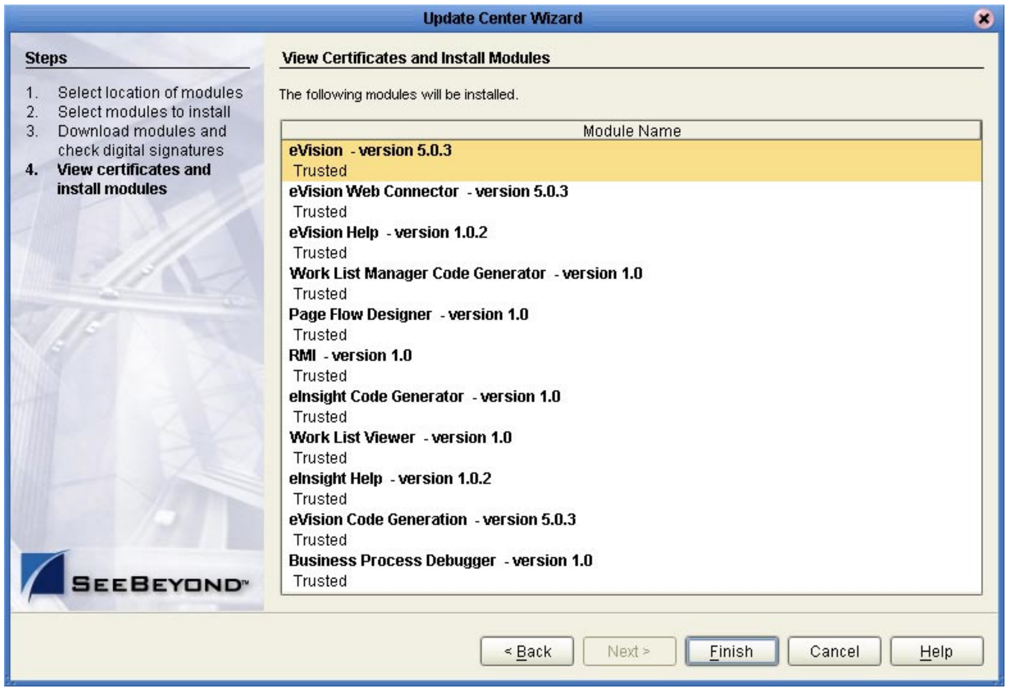
Download In Progress ...

When the download is complete, the message **Done** appears below the progress bars.

7 Click **Next**.

8 The Update Center Wizard displays the list of certificates and installed modules. See Figure 8.

**Figure 8** Update Center Wizard: View Certificates and Install Modules


Update Center Wizard
Steps
1. Select location of modules
2. Select modules to install
3. Download modules and check digital signatures
4. View certificates and install modules
SEEBEYOND
< Back
Next >
Finish
Cancel
Help

View Certificates and Install Modules

The following modules will be installed.

| Module Name |
|---|
| eVision - version 5.0.3<br>Trusted |
| eVision Web Connector - version 5.0.3<br>Trusted |
| eVision Help - version 1.0.2<br>Trusted |
| Work List Manager Code Generator - version 1.0<br>Trusted |
| Page Flow Designer - version 1.0<br>Trusted |
| RMI - version 1.0<br>Trusted |
| eInsight Code Generator - version 1.0<br>Trusted |
| Work List Viewer - version 1.0<br>Trusted |
| eInsight Help - version 1.0.2<br>Trusted |
| eVision Code Generation - version 5.0.3<br>Trusted |
| Business Process Debugger - version 1.0<br>Trusted |

9 Click **Finish**. The **Restart the IDE** dialog box appears.

10 Click **OK**.

When you log into Enterprise Designer again, you can start using eVision Studio.

## 3.3 IBM AIX Configuration Changes

If you using a Repository that is running IBM AIX, you must perform the following procedure in order to monitor eVision Studio Projects in the ICAN Monitor.

**To make the configuration changes on an IBM AIX Repository**

1. Go to the computer on which the Repository is installed.
2. If the Repository is running, shut it down.
3. Set the **DISPLAY** environment variable to ***somehost*:0.0**, where ***somehost*** is the hostname or IP address of one of the computers that will be using the ICAN Monitor. The UNIX user that starts the Repository must perform this step.

   Here is an example for the csh shell:

   ```
   setenv DISPLAY 10.1.192.13:0.0
   ```

   Here is an example for the sh shell:

   ```
   DISPLAY=10.1.192.13:0.0
   export DISPLAY
   ```
4. Open the **startserver.sh** file in the ***ICAN-root*/repository** directory.
5. Add the following command to the **JAVA_OPTS** environment variable:

   ```
   -Djava.awt.headless=true
   ```
6. Save the file.
7. Start the Repository.

Chapter 4

# Using the Page Layout Designer

This chapter focuses on eVision Studio's Page Layout Designer. Each pre-built component that you can include in a Page Layout is described. This chapter also describes how to create a Page Link.

**This Chapter Includes:**

## 4.1 Page Layout Designer Overview

Through the use of familiar drag-and-drop techniques and text-based property sheets, the Page Layout Designer allows you to interactively add graphics, text, and programmatic content to the Web pages in your application.

You drag pre-built components from a component palette and position them on the design canvas. The component's property sheet is opened automatically. The property sheet allows you to specify attributes, such as the component's logical name and the user-facing text to be displayed on a Web page.

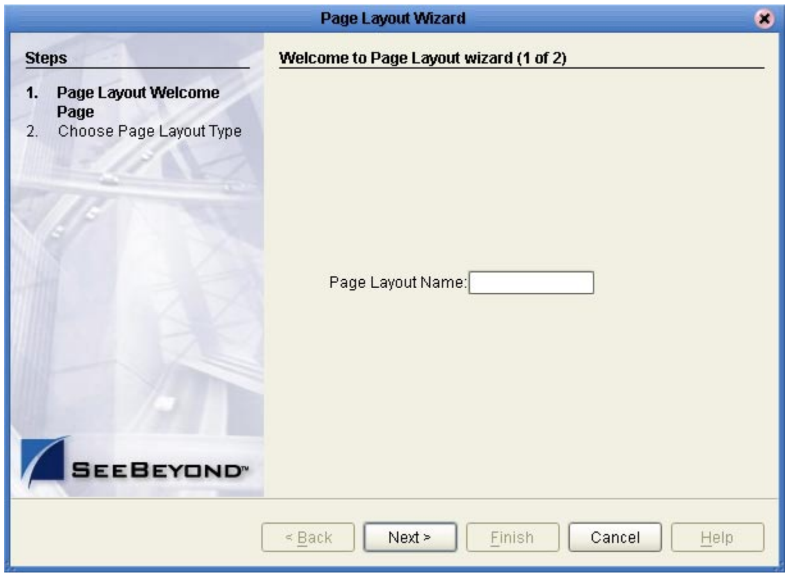
As you create your Web pages, you can preview the design in your browser at any time.

**To add a Page Layout to a Project**

1. In the Project Explorer, right-click the Project.
2. On the context menu, click **New**, and then select **Page Layout**.

   Step 1 of the Page Layout Wizard appears. See Figure 9.

**Figure 9** Page Layout Wizard - Step 1


Page Layout Wizard
Steps
1. Page Layout Welcome Page
2. Choose Page Layout Type
Welcome to Page Layout wizard (1 of 2)
SEEBEYOND
< Back
Next >
Finish
Cancel
Help

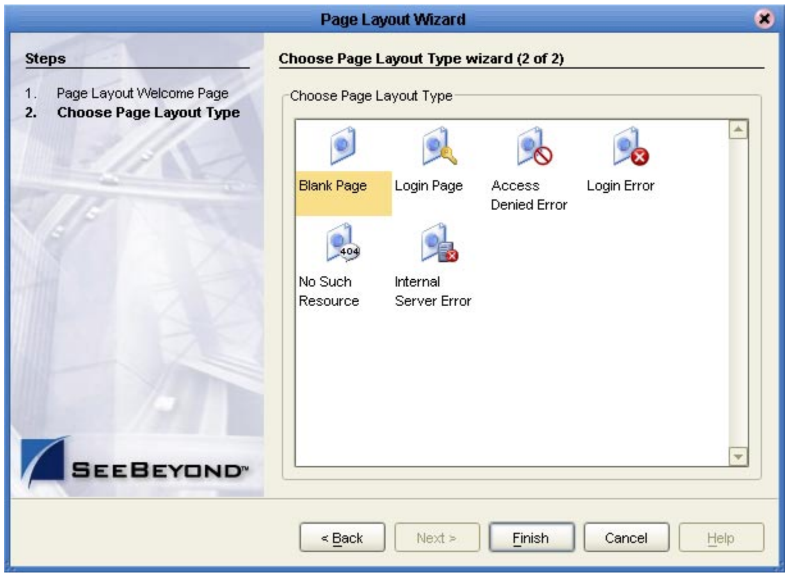
Page Layout Name:

3 In the **Page Layout Name** field, type a unique name for the new Page Layout.

4 Click **Next**.

Step 2 of the Page Layout Wizard appears. See Figure 10.

**Figure 10** Page Layout Wizard - Step 2



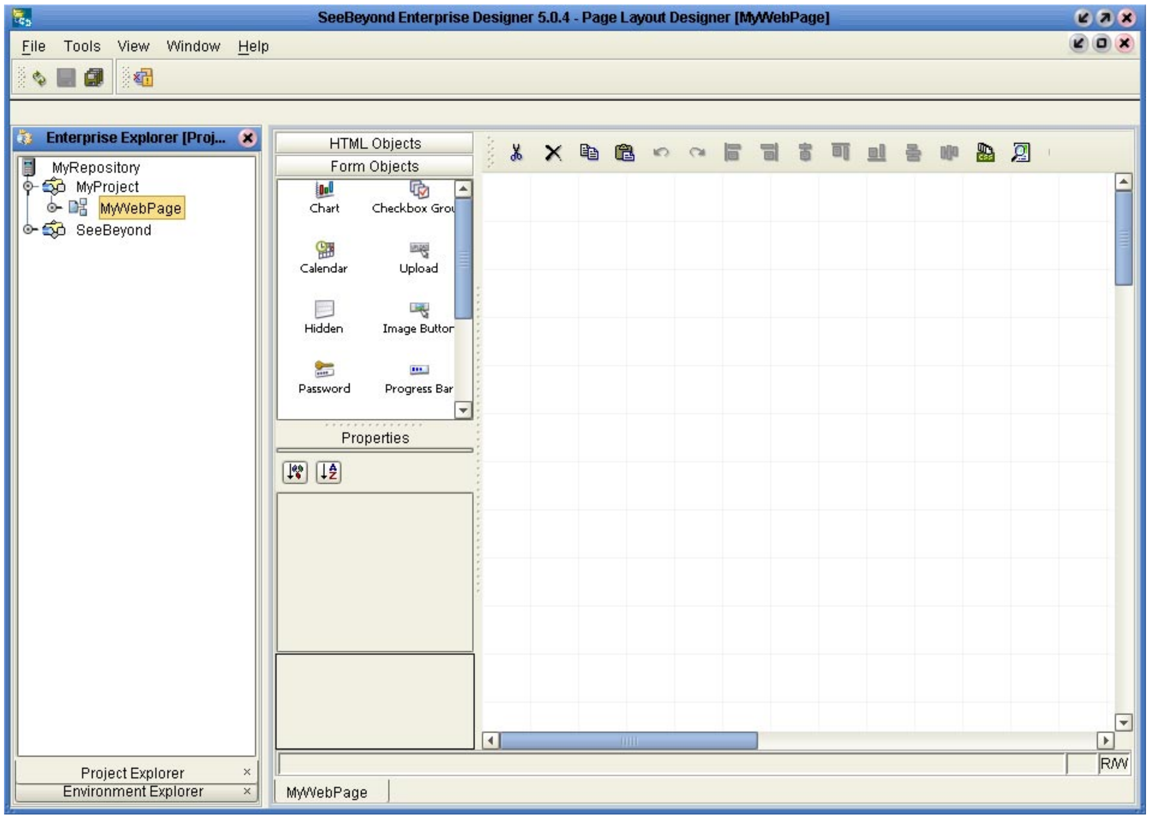
5 Click one of the Page Layout types based on the Project's requirements.

***Note:*** *Chapter 9, **"Authentication and Error Handling"** describes the Login Page, Access Denied Error, Login Error, No Such Resource, and Internal Server Error types.*

6 Click **Finish**.

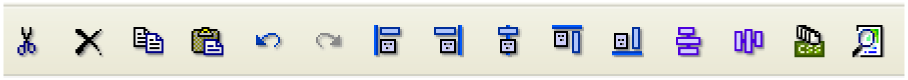
The Page Layout is added to the Project Explorer, and the Page Layout Designer appears with a blank canvas. See Figure 11.

**Figure 11** Page Layout Designer



The Page Layout Designer toolbar allows you to manipulate the objects that you place on the canvas. The toolbar is shown in Figure 12.

**Figure 12** Page Layout Designer Toolbar



Each tool is described in Table 2.

**Table 2** Page Layout Designer Tools

| Tool | Name | Purpose |
|---|---|---|
| ✂ | Cut Components | Cuts a component from the canvas and places it on the clipboard. |
| ✕ | Delete | Deletes a component permanently. |
| ⧉ | Copy Components | Copies a component from the canvas and places it on the clipboard. |
| 📋 | Paste Components | Pastes a component from the clipboard onto the canvas. |
| ↶ | Undo | Rolls back your most recent input or interaction, and then continues backward up to the last time the layout was saved. |
| ↷ | Redo | Reverses the most recent rollback, and continues forward up to your last input or interaction. |
|  | Left Align | Aligns two or more selected components with the left-most component in the selected group. |
|  | Right Align | Aligns two or more selected components with the right-most component in the selected group. |
|  | Center Align | Aligns two or more selected components with the center of the canvas (the center of the Web page). |
|  | Top Align | Aligns two or more selected components with the top-most component in the selected group. |
|  | Bottom Align | Aligns two or more selected components with the bottom component in the selected group. |
|  | Vertical Spacing | Creates equal vertical spacing between objects in a group of three or more, based on an averaging algorithm. |
|  | Horizontal Spacing | Creates equal horizontal spacing between objects in a group of three or more, based on an averaging algorithm. |

| Tool | Name | Purpose |
|---|---|---|
|  | Edit CSS | Launches the **Choose CSS to Edit** dialog box, which allows you to select an imported style sheet to edit with the eVision Style Editor. For more information, see **"Style Sheets" on page 57**. |
|  | Preview | Previews the Web page layout in your browser. |

## 4.2 Graphical User Interface Components

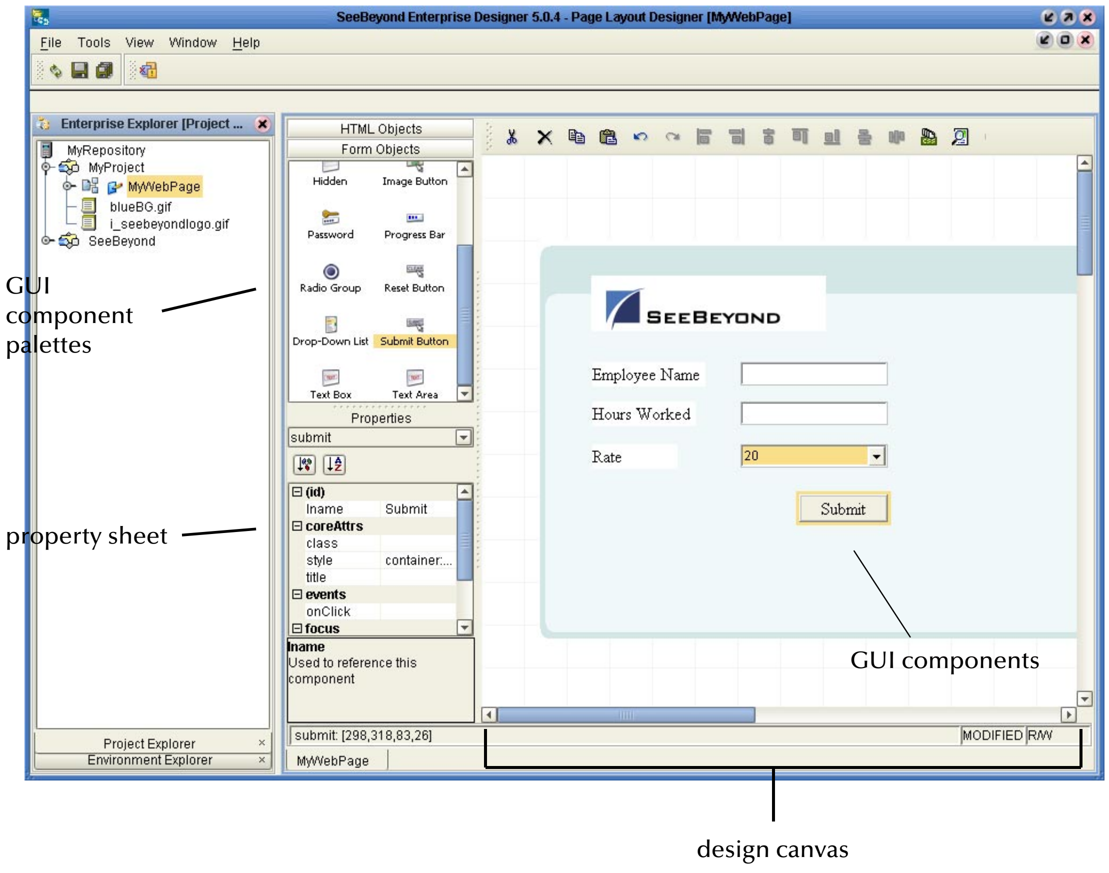
Graphical User Interface (GUI) components gather input data from the user at run-time, and add functions and interactivity to Web pages.

To add a GUI component to the canvas, you select a component from the **HTML Objects** or **Form Objects** palette, drag it onto the canvas, and release the mouse button. The Page Layout Designer provides familiar Web-centric design elements, such as check boxes, text boxes, drop-down lists, graphics containers, horizontal lines, and submit buttons.

GUI components are pre-built combinations of Java classes and JSP code that represent Web interface elements. When you place a GUI component on the canvas, the component's property sheet is opened automatically. In the property sheet, you add the labeling, functionality, and appearance attributes that you want the component to have when it is presented to the user in a browser.

In the Page Layout Designer, the upper left pane contains the GUI component palettes. The lower left pane contains the property sheets. See Figure 13.

**Figure 13** GUI Component Palettes and Property Sheet



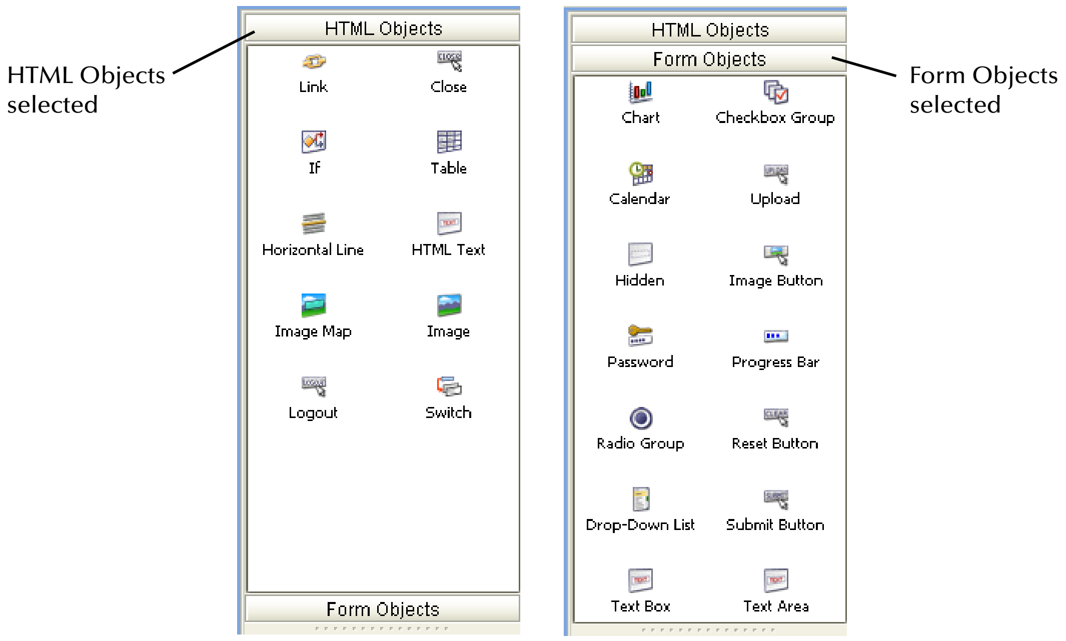
### 4.2.1 GUI Component Palettes

The Page Layout Designer provides two palettes that allow you to drag and drop GUI components onto the canvas to quickly create a Web page layout from eVision's collection of pre-built objects.

When you place a GUI component on the canvas, you then follow up by customizing its functional and visual presentation properties in the **Properties** window.

To access the **HTML Objects** palette, select the **HTML Objects** title bar. To access the **Form Objects** palette, select the **Form Objects** title bar. See Figure 14.

**Figure 14** HTML and Form Objects Palettes



## HTML Objects

The **HTML Objects** palette allows you to drag and drop HTML-based GUI components onto the Page Layout Designer canvas. HTML objects are represented graphically on a Web page and may or may not have programmatic attributes. HTML objects are described in Table 3.

**Table 3** HTML Objects

| Component | Name | Purpose |
|---|---|---|
| | Link | Creates a link to another location, another page, or an external Web site. Page links are created using the Page Link Wizard. For more information, see **"Linking Pages" on page 65**. |
| | Close | Allows the user to close the browser window. The Page Flow is automatically ended.<br><br>For more information, see **"Close Buttons" on page 46**. |
| If | If | Allows you to conditionally display an area of a Page Layout.<br><br>If the **value** property is set to **true**, then the content inside this component (for example, a company logo) will appear at runtime. If the **value** property is set to **false**, then the content will not appear at runtime. |

| Component | Name | Purpose |
|---|---|---|
| | Table | Creates a table of rows and columns. Table cells may contain any object from the HTML or Forms Objects palettes. A table can be static or dynamic. For more information, see **"Tables" on page 41** and **"Dynamic Tables" on page 42**. |
| | Horizontal Line | Places a horizontal line on the canvas. Use the Horizontal Line object to create visual separations in your layouts.<br><br>The line is initially of a fixed length. To extend the line, place the pointer on the end that you want to change (right or left, up or down). The pointer will change to a "resize" arrow (**<-->**). Drag the line end to modify the length. The bidirectional arrow cursor is dual-purpose. You can add thickness to the line as well as adjust the length. Drag the line end carefully, taking care not to add thickness to the line.<br><br>To move the line, click the line's mid-point (avoid the end points), and drag it to a new location. |
| | HTML Text | Creates a field on the canvas to hold HTML (static) text, or can act as a placeholder for dynamic text. You use this component for labels and general information on a page. |
| | Image Map | Allows you to create a "hotspot" on an image that a user can select to perform a link action, linking to another page or an area within the current page. You import the base image the same way that you import a standard graphic image.<br><br>**Note:** Images must reside in your Project in the Repository before you can access them.<br><br>For more information, see **"Image Maps" on page 44**. |
| | Image | Holds a static graphic image. When you drag the icon onto the canvas, you are prompted to select the image.<br><br>**Note:** Images must reside in your Project in the Repository before you can access them.<br><br>For more information, see **"Importing Images" on page 50**. |
| | Logout | Allows the user to exit the eVision Web application. Clicking this button returns the user to the beginning of the Page Flow.<br><br>For more information, see **Logout Buttons** on page 47. |

| Component | Name | Purpose |
|---|---|---|
| | Switch | Allows you to conditionally display an area of a Page Layout.<br><br>You can specify two or more cases and add different content to each case. To add a case, right-click the component and select **Add Case**. The **case** property specifies the name of the case that appears at runtime. |

## Form Objects

Form Objects are pre-built combinations of Java classes and JSP code representing Web interface entities. These objects allow users to communicate with Page Flows in the run-time environment. Form objects are described in Table 4.

**Table 4** Form Objects

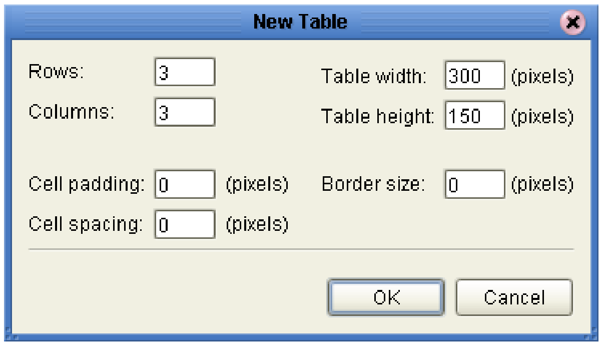
| Component | Name | Purpose |
|---|---|---|
| | Chart | Creates a two-dimensional chart. eVision provides a variety of predefined chart types, including area charts, bar charts, line charts, and pie charts.<br><br>For more information, see Chapter 8, **"Creating Charts"**. |
| | Checkbox Group | Creates a group of check boxes. Indicates inclusive user selection at run-time. You can create multiple check boxes within a group.<br><br>For more information, see Check Box Groups on page 47. |
| | Calendar | Adds a calendar under a drop-down arrow. On a Web page, selecting a date on the calendar object populates an attached text box with the selected date. |
| | Upload | Allows the user to upload files and data to an application. |
| | Hidden | Creates a hidden text field (a text field that is hidden from users at run-time), which can be used to pass session information to another page without being visible to users. |
| | Image Button | Creates an image "hotspot" that the user can click to perform an action, such as launching an application or jumping to another internal Web page or an external Web site. |
| | Password | Creates a password input box with asterisks that mask the password. |

| Component | Name | Purpose |
|---|---|---|
|  | Progress Bar | Creates a progress bar that allows the user to monitor the progress of a particular operation on a Web page. |
| ◉ | Radio Group | Switches an attribute or condition on or off. You can create multiple buttons in a group.<br><br>**Note:** The **value** property cannot be empty. This object requires that a value be assigned in order to pass the true/false condition.<br><br>For more information, see **Radio Groups** on page 48. |
| CLEAR | Reset Button | Allows the user to cancel an operation or reset values to a default condition. |
|  | Drop-Down List | Allows the user to select an entry from a drop-down list. You can create multiple entries on a list.<br><br>For more information, see **Drop-Down Lists** on page 49. |
| SUBMIT | Submit Button | Allows the user to launch an operation or submit text to the application. Passes accumulated values to the ICAN system. |
| TEXT | Text Box | Allows the user to type text that will be displayed on the Web page surrounded by a bounding box. |
| TEXT | Text Area | Allows the user to type an extensive body of text and place it anywhere on the Web page without disturbing formatting. |

## 4.2.2 Tables

When you drag the **Table** component onto the canvas, the **New Table** dialog box appears. This dialog box allows you to specify basic properties, such as the number of rows and columns. See Figure 15.

**Figure 15** New Table Dialog Box



Rows: 3
Columns: 3
Table width: 300 (pixels)
Table height: 150 (pixels)
Cell padding: 0 (pixels)
Cell spacing: 0 (pixels)
Border size: 0 (pixels)

Once you create a table, you can modify the number of rows and columns in one of two ways:

- Change the **rows** and **cols** properties in the **Properties** sheet
- Right-click a table cell and select **Grow** or **Shrink**

To add a background image to a table, import the image into the Project, select the entire table, and modify the **background** property. To remove a background image, right-click a table cell and select **Clear Table Background Image**.

To add a background color to a table, select the entire table and modify the **bgColor** property. To remove a background color, set the **bgColor** property to white.

To specify the thickness of the outer border, select the entire table and modify the **border** property. To specify whether inner borders are displayed, select the entire table and modify the **rules** property.

To remove the contents of a table cell, right-click the cell and select **Remove Cell Contents**.

## 4.2.3 Dynamic Tables

*Dynamic tables* are created like static tables but with one or more rows designated as "dynamic." In a dynamic table, the total number of rows, and the row content are undefined. At run-time, table rows and content are dynamically generated by a repeating node in a Page Flow.

**To define a dynamic row in a table**

1. Select a cell in the table. The cell will be highlighted in blue.
2. Right-click the cell.
3. On the context menu, select **Set As Dynamic Row**.

   The row will be marked as being dynamic.

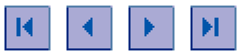
### Adding Pagination to a Dynamic Table

You can add a "page-forward, page-back", "page-first, page-last" function to a row or rows in a dynamic table. The Pagination feature allows table cell data to be incrementally displayed. For example, if the table is set to read the results of a database query, the table will display the first 10 items returned by query, and allow the user to page forward to the second 10 items, and so on.

All rows are retrieved onto the client side, and the pagination is performed on the client side.

eVision provides the following default image files for the pagination buttons: **firstImg.gif**, **prevImg.gif**, **nextImg.gif**, and **lastImg.gif**. See Figure 16.

**Figure 16** Default Pagination Buttons

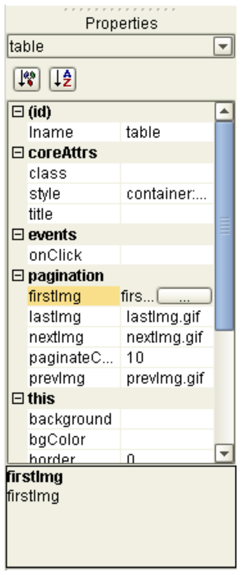


If you want to use your own image files, you must first import them into the Project. See **"Importing Images" on page 50**. You then specify the replacement image files during the following procedure.

### To add pagination to a row in a dynamic table

1. Right-click the dynamic row.
2. On the context menu, click **Paginate**.
3. Click and drag a bounding box around the table.
4. In the **Properties** sheet, the **paginateCount** property specifies how many rows the table will accept at a time. The default value is 10. If desired, increase or decrease the value.
5. In the **Properties** sheet, the **firstImg**, **lastImg**, **nextImg**, and **prevImg** properties are set to the default image files for the pagination buttons. See Figure 17.

**Figure 17** Pagination Properties



Properties

table

| | |
|---|---|
| **(id)** | |
| lname | table |
| **coreAttrs** | |
| class | |
| style | container:... |
| title | |
| **events** | |
| onClick | |
| **pagination** | |
| firstImg | firs... ... |
| lastImg | lastImg.gif |
| nextImg | nextImg.gif |
| paginateC... | 10 |
| prevImg | prevImg.gif |
| **this** | |
| background | |
| bgColor | |
| border | 0 |

**firstImg**
firstImg

If you want to use your own image files, then click a property value, navigate to the image file, and click **Open**.

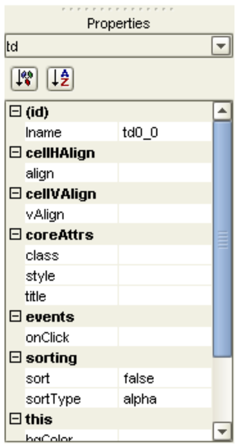
### Adding Sorting to a Dynamic Table

You can specify that a column in a dynamic table will be sorted. The sort criteria include alphabetic, numeric, and date.

**To add sorting to a dynamic table**

1. In the row immediately above the dynamic row, click the cell in the column that you want to sort.
2. In the **Properties** sheet, make sure that the **td** properties are displayed. See Figure 18.

**Figure 18** td Properties



Properties

td

| | |
|---|---|
| ⊟ **(id)** | |
| lname | td0_0 |
| ⊟ **cellHAlign** | |
| align | |
| ⊟ **cellVAlign** | |
| vAlign | |
| ⊟ **coreAttrs** | |
| class | |
| style | |
| title | |
| ⊟ **events** | |
| onClick | |
| ⊟ **sorting** | |
| sort | false |
| sortType | alpha |
| ⊟ **this** | |
| bgColor | |

3. Set the value of the **sort** property to **true**.
4. Specify how the column should be sorted by selecting a value for the **sortType** property. If you want to consider case when sorting alphabetic values, select the **alpha** value; otherwise, select the **alphaIgnoreCase** value.
5. Save the Page.

## 4.2.4 Image Maps

You create an image map by bounding an area of an image and attaching linking code to the area within the boundary. Users can then click the area to execute the code. You can define an image map to link to an external Web site, or to link to another page in the Page Flow.

***Important:*** *Before you begin, you must import the image into the Project. For more information, see* **“Importing Images” on page 50**.

When you work with image maps, the following tools are added to the Page Layout Designer toolbar:

- The **Select Link Area** tool highlights the link area.
- The **Select Image Map** tool highlights the image.

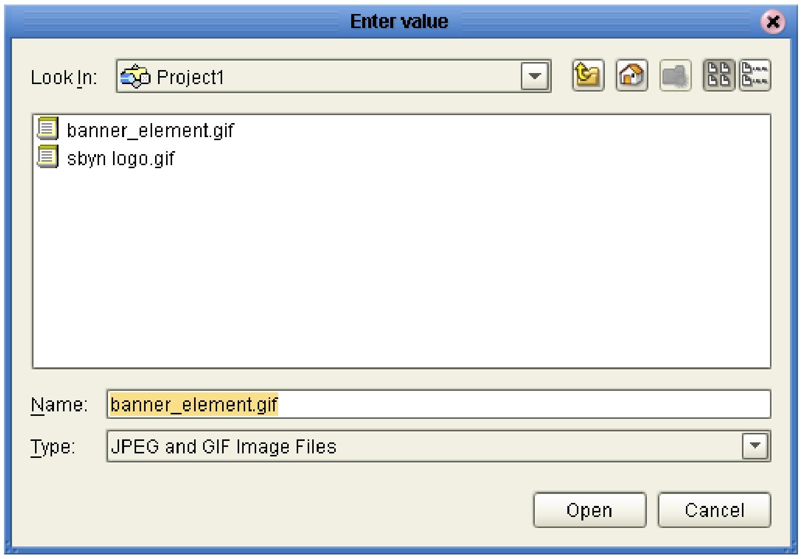
Only one of these tools is enabled at a time.

#### To create an image map

1. From the **HTML Objects** palette, drag the **Image Map** icon onto the canvas.

   The **Enter value** dialog box appears. See Figure 19.

**Figure 19** Selecting the Image for an Image Map



banner_element.gif
sbyn logo.gif

Name: banner_element.gif

Type: JPEG and GIF Image Files

2. Select the image for the image map.
3. Click **Open**.

   The image appears on the canvas. The gold outline represents the outside boundaries.

4. In the **Properties** sheet, define the value for **lname** (the component's logical name) by doing the following:

   A In the left column, select the **lname** property.

   B In the right column, delete the default value and enter a descriptive name for the component (for example, **logoImageMap**).

   **lname** is the internal, logical name for the component, which identifies the component in the Page Flow.

5. If you want to resize the image container, click and drag a highlighted boundary element. The shape of the bidirectional arrow cursor indicates which way the container will be resized.

   If you want to move the image container, click the container in the center (away from the edges) and drag it to the new location.

6. To add a link to the image, do one of the following:

   - In the Project Explorer tree, drag a Page Link and drop it onto the image. This link has predefined parameters and a target location.

- From the **HTML Objects** palette, drag the **Link** icon onto the image. This link requires that you specify link parameters and a target location.

When you drop a link onto the image, a rectangle appears. This is the *link area*.

7 In the **Properties** sheet, specify the target location of the link area (if necessary):

A In the left column, select the **href** property.

B In the right column, enter the target location (for example, **http://www.seebeyond.com**).

8 If you want to resize the link area, click and *slowly* drag a highlighted boundary element.

If you want to move the link area, click the link area in the center (away from the edges) and drag it to the new location.

You can also resize and move the link area by manually entering the coordinates in the **Properties** sheet. In the left column, click the **coords** property. In the right column, change the value.

If the link area that you are trying to size to is too small, right-click the image and choose **Set Link Area to Image Size**.

9 If desired, create additional link areas in the image.

#### To test an image map

1 On the Page Layout Designer toolbar, click the **Preview** icon.

In the preview image, when you pass the pointer over the mapped area, the cursor changes to the "hand" icon.

2 Click the defined image area to execute the link code.

### 4.2.5 Close Buttons

The **Close** button allows the user to close the browser window. The Page Flow is automatically ended.

#### To create a Close button

1 From the **HTML Objects** palette, drag the **Close** icon onto the canvas.

The **Close** button appears. The gold outline represents the outside boundaries of the button.

2 In the **Properties** sheet, change the default value of the **lname** property (the component's logical name).

3 If you want to change the default image, do the following:

A Import the image that you want to use into your Project. For more information, see **Importing Images** on page 50.

B In the **Properties** sheet, click the **src** property.

C Place your cursor over the existing value and click the **Command** button (...). The **Enter value for "src"** dialog box appears.

D Navigate to the image that you want to use.

E Click **Open**.

## 4.2.6 Logout Buttons

The **Logout** button allows the user to exit the eVision Web application. Clicking this button returns the user to the beginning of the Page Flow. You can place a **Logout** button on any page.

### To create a Logout button

1 From the **HTML Objects** palette, drag the **Logout** icon onto the canvas.

The **Logout** button appears. The gold outline represents the outside boundaries of the button.

2 In the **Properties** sheet, change the default value of the **lname** property (the component's logical name).

3 If you want to change the default image, do the following:

A Import the image that you want to use into your Project. For more information, see **Importing Images** on page 50.

B In the **Properties** sheet, click the **src** property.

C Place your cursor over the existing value and click the **Command** button (...). The **Enter value for "src"** dialog box appears.

D Navigate to the image that you want to use.

E Click **Open**.

## 4.2.7 Check Box Groups

The **Checkbox Group** component allows you to create a group of check boxes. You specify the label and value for each check box.

### To create a Checkbox Group

1 From the **Form Objects** palette, drag the **Checkbox Group** component onto the canvas.

2 Right-click the component.

3 On the context menu, select **Edit Options**.

The **Edit Options** dialog box appears.

4 Select a **Label** field and type a label.

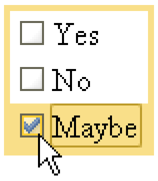
The label will be displayed next to the check box at run-time.

5 In the **Value** field, type a value.

The value will be submitted when the user checks the box at run-time.

6 To add additional check boxes to the group, click **Add** and repeat the previous steps. See Figure 20.

**Figure 20** Editing Checkbox Group Options



7 Click **OK**.

**Note:** *You can drag a **Checkbox Group** component into a table cell and make the table row dynamic. At run-time, the component can be modified by external sources to generate additional cells containing check boxes.*

## 4.2.8 Radio Groups

The **Radio Group** component allows you to create a group of radio buttons. You specify the label and value for each radio button.

**To create a Radio Group**

1 From the **Form Objects** palette, drag the **Radio Group** component onto the canvas.

2 Right-click the component.

3 On the context menu, select **Edit Options**.

The **Edit Options** dialog box appears.

4 Select a **Label** field and type a label.

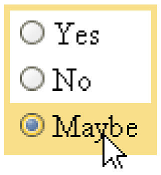
The label will be displayed next to the button at run-time.

5 In the **Value** field, type a value.

This value will be submitted when the user clicks the button at run-time.

6 To create additional buttons in the group, click **Add** and repeat the previous steps. See Figure 21.

**Figure 21** Editing Radio Group Options



7 Click **OK**.

**Note:** *You can drag a **Radio Group** component into a table cell and make the table row dynamic. At run-time, the component can be modified by external sources to generate additional cells containing radio buttons.*

### 4.2.9 Drop-Down Lists

The **Drop-Down List** component allows you to create multiple user-selectable options. You specify the label and value for each option.

You can make a **Drop-Down List** component dynamic. Dynamic components are extended or replicated using the Business Rule Designer.

1 From the **Form Objects** palette, drag the **Drop-Down List** component onto the canvas.

2 Right-click the component.

3 On the context menu, select **Edit Options**.

The **Edit Options** dialog box appears.

4 Select a **Label** field and type a label.

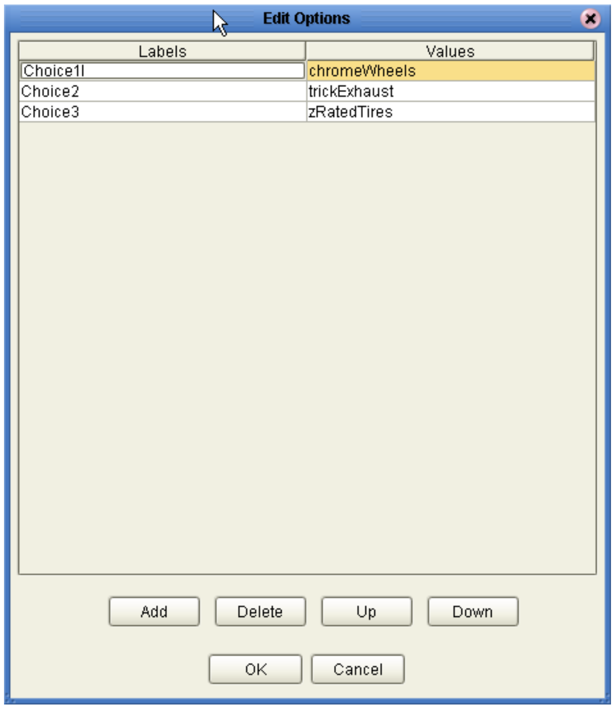
The label will be displayed as an option to users at run-time.

5 In the **Value** field, type a value.

The value will be submitted when the user clicks the label at run-time.

6 To create additional **Drop-Down List** entries, click **Add** and repeat the previous steps. See Figure 22.

**Figure 22** Edit Options Dialog Box

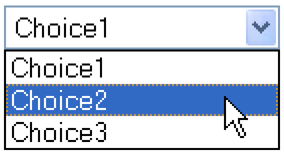


| Labels | Values |
|---|---|
| Choice1l | chromeWheels |
| Choice2 | trickExhaust |
| Choice3 | zRatedTires |

7 Click **OK**.

8 To see how the list will be presented to users at run-time, click the **Preview** icon. See Figure 23.

**Figure 23** Drop-Down List Preview



9 To make the **Drop-Down List** component dynamic, do the following:

A Right-click the component.

B On the context menu, select **Make Dynamic**.

At run time, a dynamic **Drop-Down List** component can be extended with additional labels and values generated by external sources at run time. The **Edit Options** menu is disabled when you select **Make Dynamic**.

***Note:*** *The values of dynamic components do not appear when you click the* ***Preview*** *icon, because the values are assigned at run time.*

## 4.2.10 Submit Buttons

The **Submit Button** component is often used for a scenario in which you want to send data entered by the user to a back-end system and then end the browser session.

To help ensure that the data is successfully sent to the back-end system, SeeBeyond recommends that you create two Page Layouts.

- The first Page Layout contains the **Submit Button** component.
- The second Page Layout is a confirmation page. Add a message such as the following:

  ```
  Your data has been submitted.
  ```

  Below the message, add a **Close** button, which enables the user to close the browser window.

An alternative approach that requires only one Page Layout is to add a **Link** component to the Page Layout that submits the data. In the **href** property for the **Link** component, enter JavaScript code that closes the browser window. With this approach, you do not need to create a confirmation page.

## 4.2.11 Importing Images

To make graphic images accessible to your Web page, you must first import them into your Project from their location on disk.

**To import an image**

1 In the Project Explorer, right-click the Project icon.

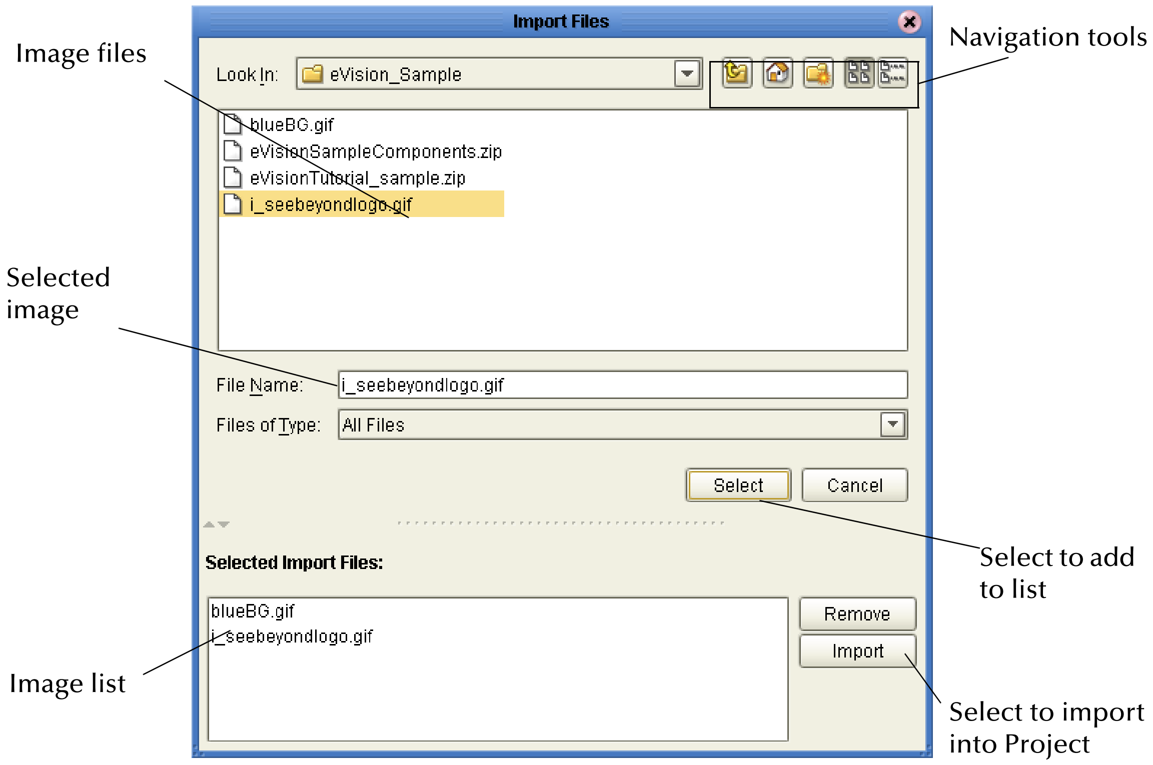
2 On the context menu, select **New**, and then select **File**.

The **Import Files** dialog box appears.

3 Navigate to the directory that contains the file or files that you want to import. See Figure 24.

The image files can reside anywhere on your machine or a network.

**Figure 24** Importing Images into a Project



4 For each file that you want to import, select the file and click **Select**.

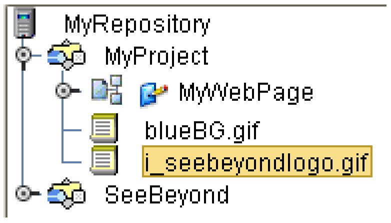
The file names appear in the **Selected Import Files** box.

5 To remove a file from the list, select the file name in the **Selected Import Files** box and click **Remove**.

6 When you are done, click **Import**.

The images are displayed under your Project in the Project Explorer tree. See Figure 25.

**Figure 25** Imported Images in Project Explorer Tree



## 4.3 Defining Component Properties

The **Properties** window displays the property sheet of a GUI component or of the overall Page Layout. The property sheet allows you to customize the behavior of the component or Page Layout.

The upper portion of the **Properties** window shows the active property type. The property type for the overall Page Layout is **page**. Each component has one or more property types. When a component is selected on the canvas, the **Properties** drop-down arrow enables you to change the property type. See Figure 26.

**Figure 26** Changing the Property Type

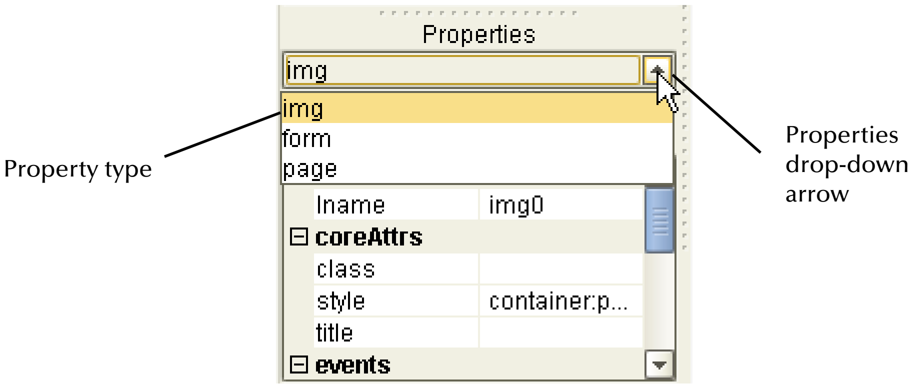
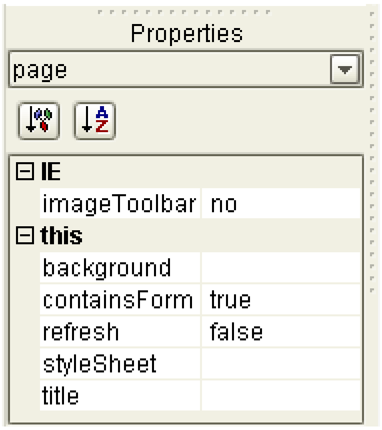


Figure 27 shows an example of the **page** property type.

**Figure 27** Page Properties



| | |
|---|---|
| IE | |
| imageToolbar | no |
| this | |
| background | |
| containsForm | true |
| refresh | false |
| styleSheet | |
| title | |

Some of the properties are required, while other properties are optional. For example, each component must have a logical name (the **lname** property), whereas you can choose whether or not to specify a tooltip that will appear at runtime.

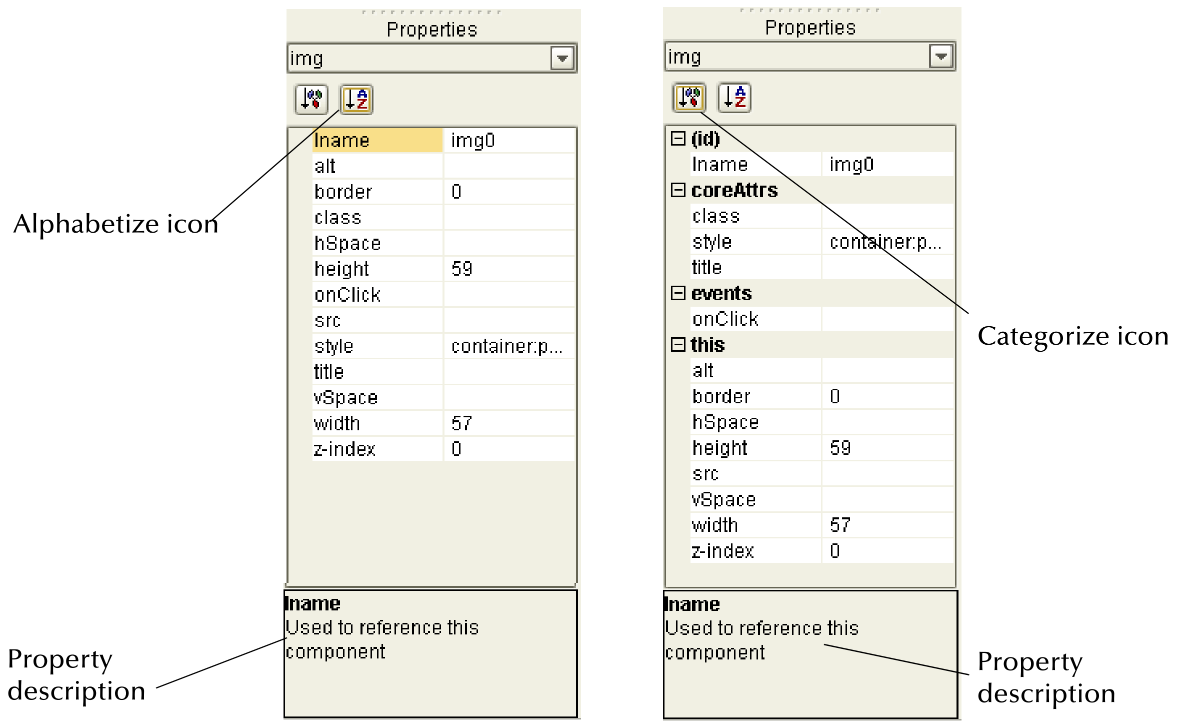
***Important:*** *Be sure to change the default value of the **lname** property to a more descriptive value. The Page Flow Designer uses the **lname** property as the identifier for* *components. Setting the **lname** property to a descriptive value will make it easier for you to identify the component in the Page Flow Designer.*

A description of each property is provided in two ways:

- In a property description box at the bottom of the **Properties** window
- As a tooltip when you place the mouse pointer over a property name

You can specify whether to display the list of properties in alphabetical or category format. For alphabetical format, click the **Alphabetize** icon. For category format, click the **Categorize** icon. See Figure 28.

**Figure 28** Swapping Component Properties Formats



img

| | |
|---|---|
| lname | img0 |
| alt | |
| border | 0 |
| class | |
| hSpace | |
| height | 59 |
| onClick | |
| src | |
| style | container:p... |
| title | |
| vSpace | |
| width | 57 |
| z-index | 0 |

**lname**
Used to reference this component

img

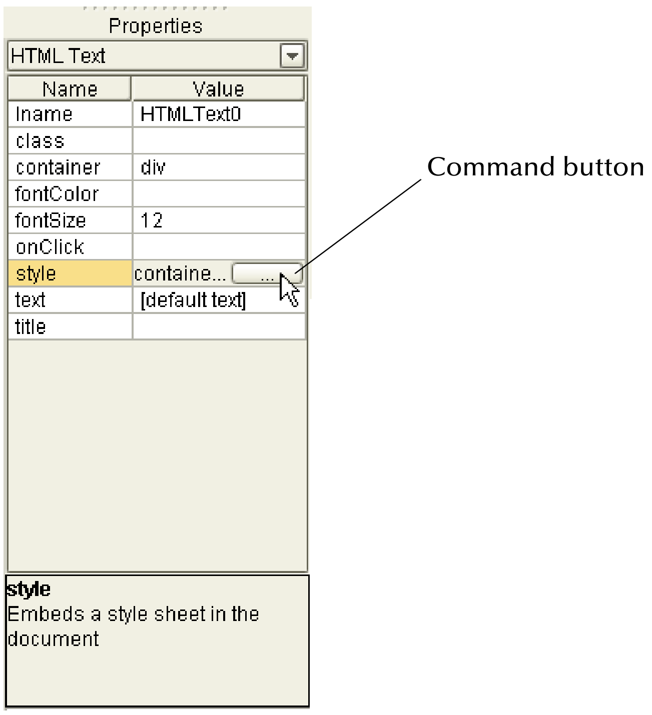
| | |
|---|---|
| ⊟ **(id)** | |
| lname | img0 |
| ⊟ **coreAttrs** | |
| class | |
| style | container:p... |
| title | |
| ⊟ **events** | |
| onClick | |
| ⊟ **this** | |
| alt | |
| border | 0 |
| hSpace | |
| height | 59 |
| src | |
| vSpace | |
| width | 57 |
| z-index | 0 |

**lname**
Used to reference this component

To define the value for a property, select the property in the left column and then do one of the following in the right column:

- Enter a value.
- Select from a drop-down list.
- Click the **Command** button (**...**). In the dialog box that appears, enter text or specify options. See Figure 29 and Figure 30.

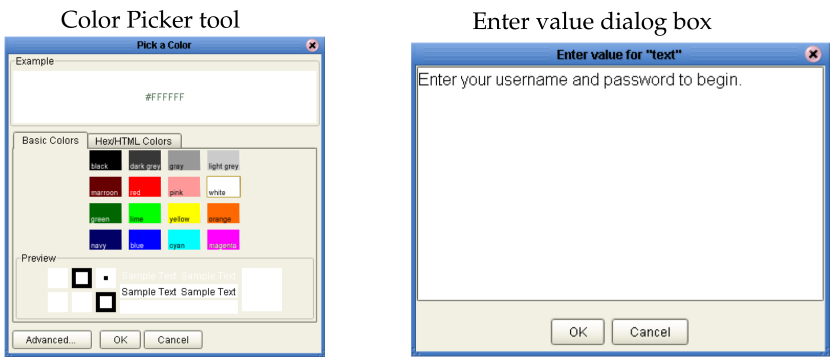
**Figure 29** Command Button



| Name | Value |
|---|---|
| lname | HTMLText0 |
| class | |
| container | div |
| fontColor | |
| fontSize | 12 |
| onClick | |
| style | containe... ... |
| text | [default text] |
| title | |

**style**
Embeds a style sheet in the document

**Figure 30** Example Dialog Boxes Displayed With the Command Button



Enter your username and password to begin.

## 4.4 Moving Overlapping Components in the Z-Direction

The standard 2D graphical orientation consists of two values: **x** (horizontal) and **y** (vertical). In 2D, a third value (**z**) implies another planar orientation, which is toward or away from the viewer (backward/forward). The z-value allows you to place objects under or on top of each other using an integer to specify the "plane."

Components can often overlap. Sometimes this is desirable; other times, it is not. You can move components in the z-direction one level at a time, forward or back, using the context menu.

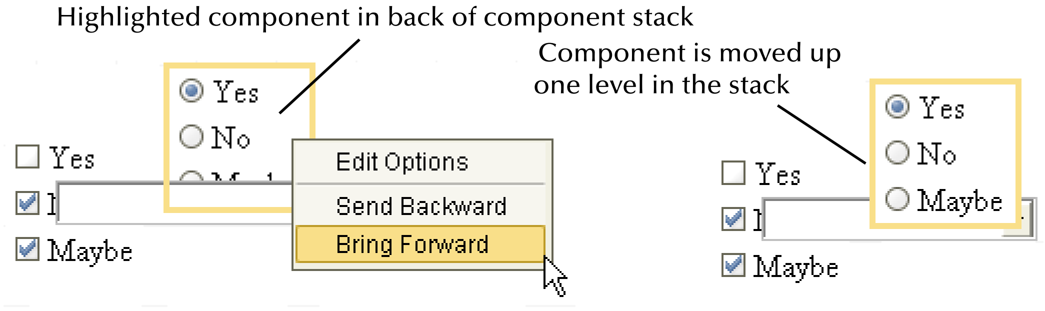
To move a component in the z-direction, one level at a time, do the following:

1 Highlight the component you want to move.

2 Place the cursor inside the component's highlight boundary and right-click the mouse.

3 On the context menu, select **Bring Forward** or **Send Backward** as appropriate. See Figure 31.

**Figure 31** Moving Components in the Z-direction


Highlighted component in back of component stack
Component is moved up one level in the stack
Yes
No
Yes
Maybe
Edit Options
Send Backward
Bring Forward
Yes
No
Maybe
Yes
Maybe

To modify the z-value of a component, right-click the component, and in the context menu, do the following:

- To move the component back (down) one level, click **Send Backward**.
- To move the component forward (up) one level, click **Bring Forward**.

You can also move an object forward or backward in the stack order by entering a negative or positive integer in the **z-index** property on the property sheet. Initially, all objects are dropped onto the same layer, where the default plane is "**0**."

- To move an image down (back) one or more levels, click the **z-index** property on the property sheet and type a negative integer; for example: **-9**.
- To move an image up (forward) one or more levels, click the **z-index** property on the property sheet and type a positive integer; for example **7**. Do not include a plus sign.

## 4.5 Manipulating Objects on the Canvas

When you are viewing a page in Preview mode, some components may not be located exactly where you want them, or a component may be too large or too small. For example, if two horizontal lines are not of equal length, and they are supposed to be separate but identical, this may detract from the overall composition of the page layout. This section describes the various ways that you can manually move and resize page components.

## 4.5.1 Moving a Single Object

To move a single object to any location, select the object. When the object is active, it will be highlighted with a color border. You can click-and-drag the object to any location on the page using the mouse. When the object is where you want it, release the mouse button.

You can also move an object using the arrow keys on your computer keyboard. When the object is where you want it, click the canvas in a blank area.

## 4.5.2 Moving a Temporary Group of Objects

To choose two or more objects to be moved simultaneously, do one of the following:

- Select the canvas in a blank area adjacent to the objects that you want to move, and drag the cursor over both objects. Your mouse movement will trace a bounding box for visual reference. You can surround the objects that you want to move, or more simply, one edge of the bounding box must intersect with an object to add it to the group.
- Press and hold the Shift key on your computer and sequentially select the objects that you want to move.

When the objects are selected, click-and-drag one of the objects and the others will follow.

You can also move a group of objects using the arrow keys on your computer keyboard. When the objects are where you want them, click the canvas in a blank area.

## 4.5.3 Resizing Objects

You can resize objects vertically or horizontally, depending on the attributes of the object. The following paragraph describes horizontal resizing.

Select the object or objects as described in **"Moving a Single Object" on page 56** and **"Moving a Temporary Group of Objects" on page 56**. Move the pointer over the left or right border of the object. The pointer changes to a bidirectional arrow. Click-and-drag the border of the object to expand it horizontally.

### Resizing Horizontal Lines

To resize a horizontal line, do either of the following:

- To modify a line horizontally, click the line at either end. When the line is highlighted, drag the bidirectional cursor arrow to the right or left to shrink or stretch the line in either direction. To finish, click the canvas anywhere outside the line object.
- To modify line thickness, click the line at either end. When the line is highlighted, drag the bidirectional cursor arrow diagonally or vertically to increase or decrease the line's thickness. To finish, click the canvas anywhere outside the line object.

## 4.6 Style Sheets

Style sheets control the fonts and formatting of a Web page. For example, in a style sheet, a font and its characteristics such as color and size are assigned to an HTML tag (for example, a paragraph or heading). When the tag is used, the font definition is employed and the font is displayed on the Web page according to the style elements assigned to it.

eVision Studio provides the following style sheets for immediate use: **eVision-default.css** and **eVision.css**. To apply a style sheet, see **"To apply a style sheet to your Web application" on page 60**.

Before you start designing a page, you may want to import and apply your own customized style sheet. To import a style sheet, see **"Importing a Style Sheet Into Your Project" on page 59**.

If you do not have a preferred style sheet that you want to apply, you may want to create an updated style sheet from the existing source files to manage the page format. You can create a custom style sheet done in two ways:

- Edit one of the default style sheets.
- Import a style sheet from an external source and modify it.

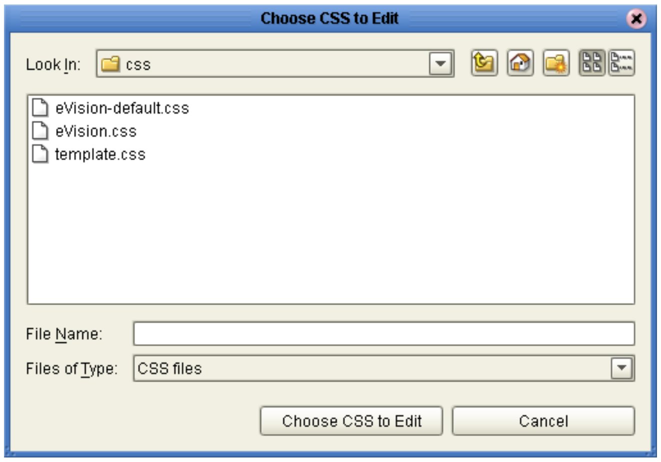
### 4.6.1 Creating a New Style Sheet from eVision.css

To create a custom style sheet, you can start by modifying one of the default style sheets and use it as a starter template.

**To create a new style sheet from eVision.css (recommended):**

1. Click the **Edit CSS** icon on the Page Layout Designer toolbar. The **Choose CSS to Edit** dialog box appears. See Figure 32.

**Figure 32** Choose CSS to Edit Dialog Box



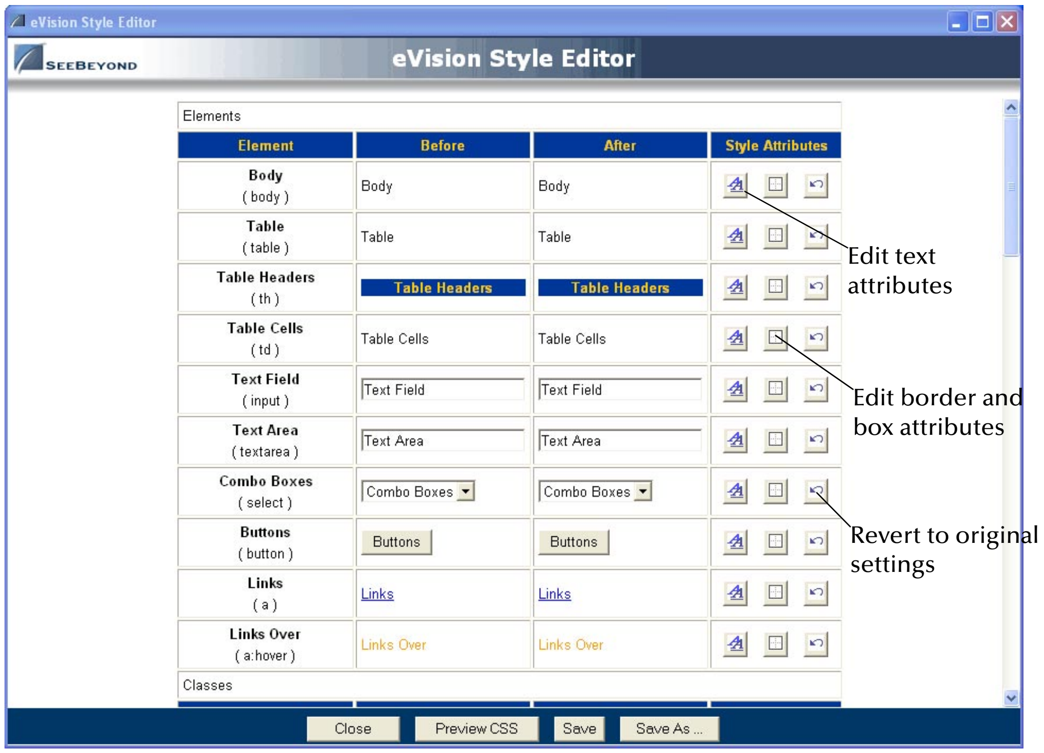
eVision-default.css
eVision.css
template.css

Files of Type: CSS files

2. Click **eVision.css**.
3. Click **Choose CSS to Edit**.

The eVision Style Editor appears. This editor allows you to modify a host of properties that will be used to manage the format of a Web page. See Figure 33.

**Figure 33** eVision Style Editor



Elements

| Element | Before | After | Style Attributes |
|---|---|---|---|
| Body ( body ) | Body | Body | |
| Table ( table ) | Table | Table | |
| Table Headers ( th ) | Table Headers | Table Headers | |
| Table Cells ( td ) | Table Cells | Table Cells | |
| Text Field ( input ) | Text Field | Text Field | |
| Text Area ( textarea ) | Text Area | Text Area | |
| Combo Boxes ( select ) | Combo Boxes | Combo Boxes | |
| Buttons ( button ) | Buttons | Buttons | |
| Links ( a ) | Links | Links | |
| Links Over ( a:hover ) | Links Over | Links Over | |

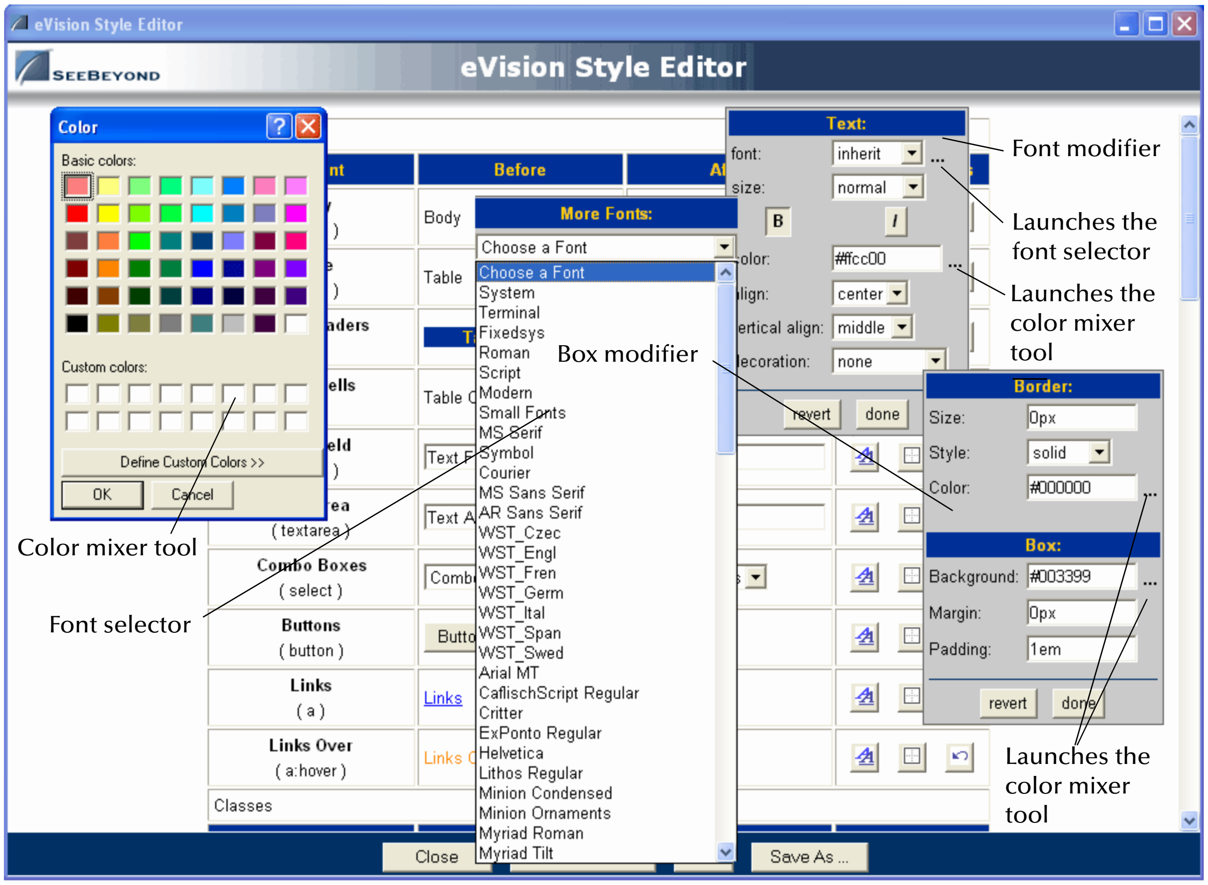
Classes

The eVision Style Editor contains a list of style sheet elements that you can modify to control the format of your Web page. Modifications that you can make to style sheet elements include:

- **Font**: type, size (percent), and color; bold and/or italic
- **Font decoration**: underscore, overscore, line-through
- **Horizontal element alignment**: left, right, center, justify
- **Vertical element alignment**: top, bottom, or center
- **Border attributes**: size, color, and line style including groove, ridge, inset, or outset
- **Box attributes**: background color, margin and padding in pixels

For a composite view of the available eVision Style Editor tools, see Figure 34.

**Figure 34** eVision Style Editor Tools



The eVision Style Editor retains the original specifications for the elements that you change. The **Before** column shows the original element configuration. The **After** column shows a real-time rendering of the elements that you changed.

To change back to the original configuration (reverse your changes), click the **revert** icon.

To view the changes you have made, click **Preview CSS**, and then click **Back to Editor**.

To save your changes and overwrite the template, click **Save**.

To save the style sheet under a new name (recommended), click **Save As** and type a unique name in the dialog box that appears. Click **OK**.

## 4.6.2 Importing a Style Sheet Into Your Project

When you are satisfied with the element configuration in your style sheet, you must import it into your Project before it can be applied.

**To import a custom style sheet**

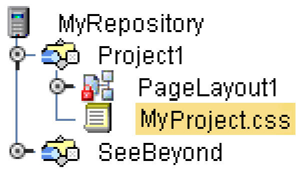
1 In the Project Explorer, right-click the Project.

2 On the context menu, click **New**, and then select **File**.

3. Navigate to the ***ICAN-root*/edesigner/usrdir/modules/ext/stc/evision_core/tigris/css** directory.
4. Select one or more style sheets and click **Select**.
5. Click **Import**.

   The style sheet(s) will appear in the Project Explorer. See Figure 35.

**Figure 35** Style Sheet in Project Explorer



The selected style sheets are now a part of your Project and can be applied your application’s Web pages.

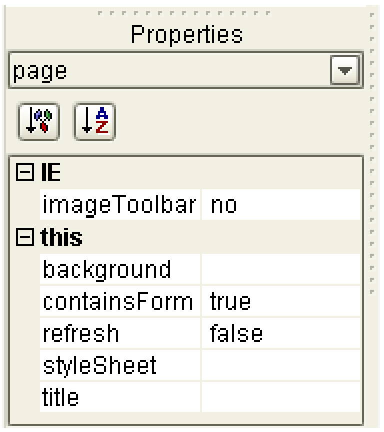
## 4.6.3 Applying a Style Sheet to Your Web Application

The final task in implementing a style sheet is to apply the style sheet to the Web pages in your Project.

**To apply a style sheet to your Web application**

1. In the **Properties** window, click the drop-down arrow.
2. On the drop-down list, click **page**. See Figure 36.

**Figure 36** Page Properties

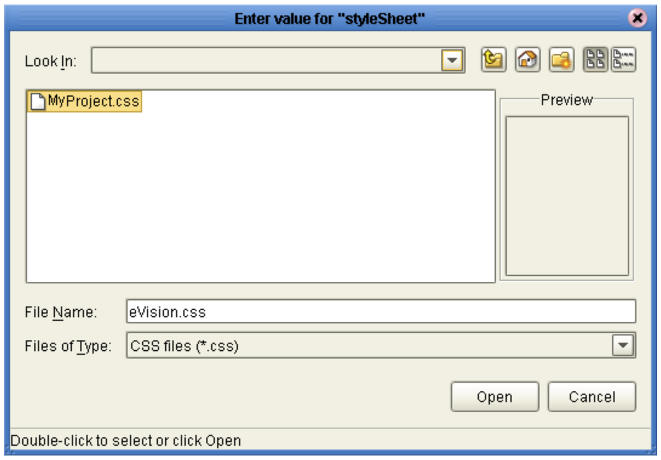


page

| | |
|---|---|
| IE | |
| imageToolbar | no |
| this | |
| background | |
| containsForm | true |
| refresh | false |
| styleSheet | |
| title | |

3. In the left column, select the **styleSheet** property.
4. In the right column, click the **Command** button (**...**). The **Enter value for “styleSheet”** dialog box appears. See Figure 37.

**Figure 37** Enter value for “styleSheet” Dialog Box



File Name: eVision.css

Files of Type: CSS files (*.css)

Double-click to select or click Open

5 Navigate to the Project folder and select the style sheet.

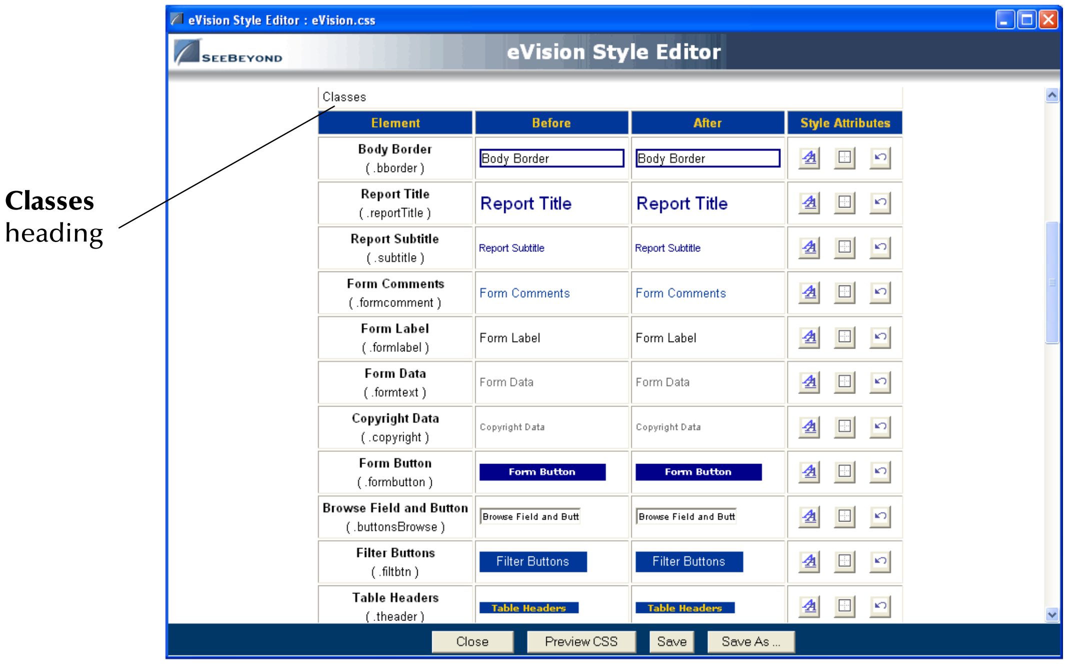
6 Click **Open**.

## 4.6.4 Applying Classes

You can also apply *classes* of specialized style elements. An *element class* is a special set of element attributes that can be applied to individual objects without disrupting the primary style sheet.

The style sheet that is packaged with eVision, **eVision.css**, provides a number of classes that are already defined and named. The default classes are accessible via the eVision Style Editor under the **Classes** heading. To see the class section in the editor, use the scroll bar and scroll down until the **Classes** heading appears. See Figure 38.

**Figure 38** eVision Style Editor - Classes



Classes

| Element | Before | After | Style Attributes |
|---|---|---|---|
| Body Border ( .bborder ) | Body Border | Body Border | |
| Report Title ( .reportTitle ) | Report Title | Report Title | |
| Report Subtitle ( .subtitle ) | Report Subtitle | Report Subtitle | |
| Form Comments ( .formcomment ) | Form Comments | Form Comments | |
| Form Label ( .formlabel ) | Form Label | Form Label | |
| Form Data ( .formtext ) | Form Data | Form Data | |
| Copyright Data ( .copyright ) | Copyright Data | Copyright Data | |
| Form Button ( .formbutton ) | Form Button | Form Button | |
| Browse Field and Button ( .buttonsBrowse ) | Browse Field and Butt | Browse Field and Butt | |
| Filter Buttons ( .filtbtn ) | Filter Buttons | Filter Buttons | |
| Table Headers ( .theader ) | Table Headers | Table Headers | |

You can modify classes when you create or modify a style sheet. Classes cannot be applied until the style sheet is imported into the Project. (See **"Importing a Style Sheet Into Your Project" on page 59**.) After importing, follow the instructions in **"Applying a Class to an Object"**.

## Applying a Class to an Object

### To apply a class to an HTML or Form object

1. On the design canvas, select the HTML or Form object to which you want to add the attributes of a class.

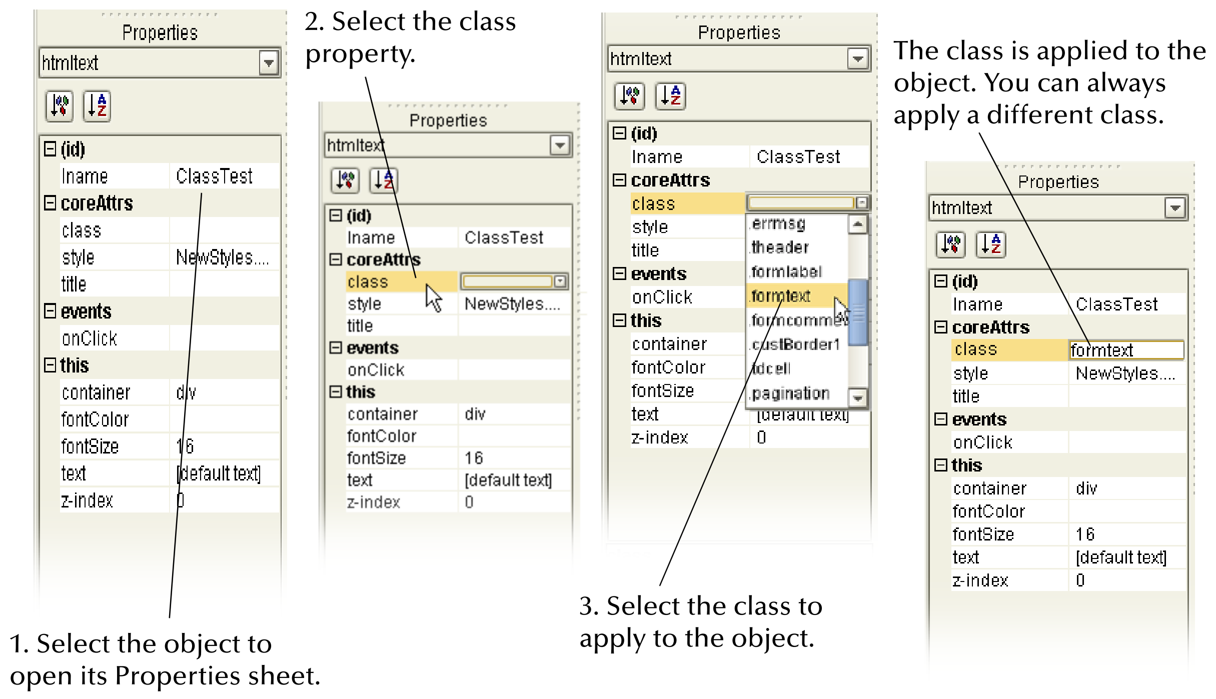
   The **Properties** sheet displays the object's properties.

2. In the **Properties** sheet, select the **class** property.

   A drop-down menu lists the classes that are available in the style sheet.

3. Select a class name from the list to apply the class to the object. See Figure 39.

**Figure 39** Mapping a Class to a Component


2. Select the class property.
The class is applied to the object. You can always apply a different class.
3. Select the class to apply to the object.
1. Select the object to open its Properties sheet.
Properties
htmltext
errmsg
theader
formlabel
formtext
formcomment
custBorder1
tdcell
pagination

| | |
|---|---|
| **(id)** | |
| Iname | ClassTest |
| **coreAttrs** | |
| class | formtext |
| style | NewStyles.... |
| title | |
| **events** | |
| onClick | |
| **this** | |
| container | div |
| fontColor | |
| fontSize | 16 |
| text | [default text] |
| z-index | 0 |

## 4.7 Automatically Refreshing Page Layouts

You can indicate that a Page Layout will automatically refresh after a specified period of time. A typical use for this feature is when the page contains dynamic content.

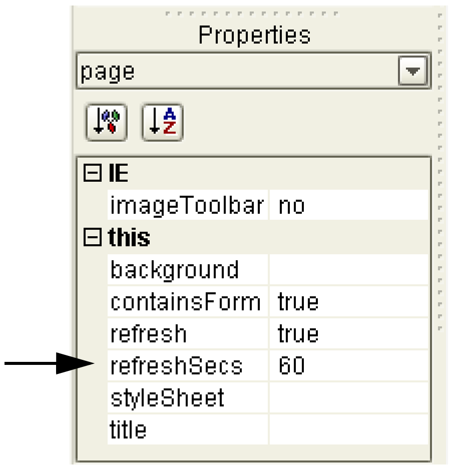
The following procedure requires you to set the **refresh** and **refreshSecs** properties of the Page Layout. If the Page Layout is *not* going to be the first Page Layout in a Page Flow, then you must set the **method** property of the immediately preceding Page Layout.

### To automatically refresh a Page Layout

1. Open the Page Layout that you want to be automatically refreshed.
2. In the **Properties** window, select the **page** property type from the drop-down menu.
3. In the left column, select the **refresh** property.
4. In the right column, set the value to **true**.

   A new property called **refreshSecs** appears. See Figure 40.

**Figure 40** refreshSecs Property



| | |
|---|---|
| ⊟ IE | |
| imageToolbar | no |
| ⊟ this | |
| background | |
| containsForm | true |
| refresh | true |
| refreshSecs | 60 |
| styleSheet | |
| title | |

5 The **refreshSecs** property indicates how many seconds to wait before reloading the page. The default value is 60 seconds. If desired, change the value.

6 Save the Page Layout.

7 If the Page Layout is *not* going to be the first Page Layout in a Page Flow, then perform the following steps.

   A Open the Page Layout that will immediately precede the current Page Layout in the Page Flow.

   B Select a Form GUI component.

   C In the **Properties** window, select the **form** property type from the drop-down menu.

   D In the left column, select the **method** property.

   E In the right column, set the value to **GET**.

   F Save the Page Layout.

## 4.8 Using the Save As Feature

The Save As feature allows you to save a Page Layout in the same Project or in a different Project.

If you save the Page Layout in the same Project, you must enter a new name for the Page Layout. If you save the Page Layout in a different Project, the Page Layout can have the same name or a different name.

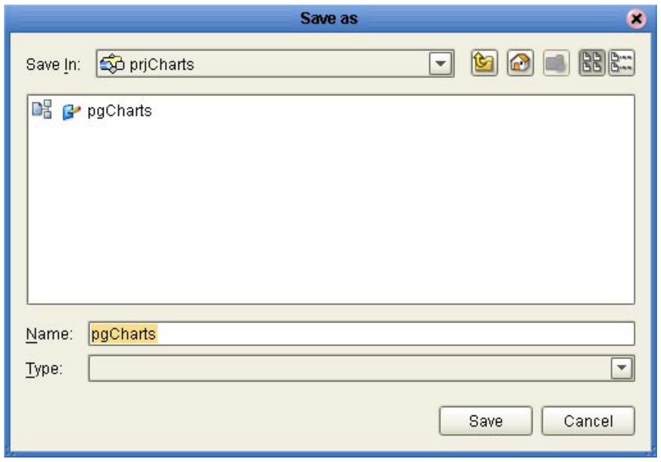
### To use the Save As feature

1 In the Project Explorer, right-click the Page Layout.

2 On the context menu, select **Save As**.

   The **Save as** dialog box appears. See Figure 41.

**Figure 41** Save as Dialog Box



3 If you want to save the Page Layout in a different Project, navigate to the desired Project.

4 If you want to change the Page Layout name, do so in the **Name** field.

5 Click **Save**.

## 4.9 Linking Pages

In eVision Web applications, links that point to internal application resources (other pages, applications, and so on), including "Home" page links, are created with the Page Link Wizard.

The Page Link Wizard allows you to:

- Identify the start (Home) page of the Page Flow
- Create text links that connect Web pages and other resources to the Page Flow
- Specify parameters to pass to other objects on destination pages

The Page Link Wizard is used to create links within the same Web application, which pass an event from a source page to a destination page. An event is any user interaction that triggers an application response; for example, selecting a radio button, or entering text in a text field.

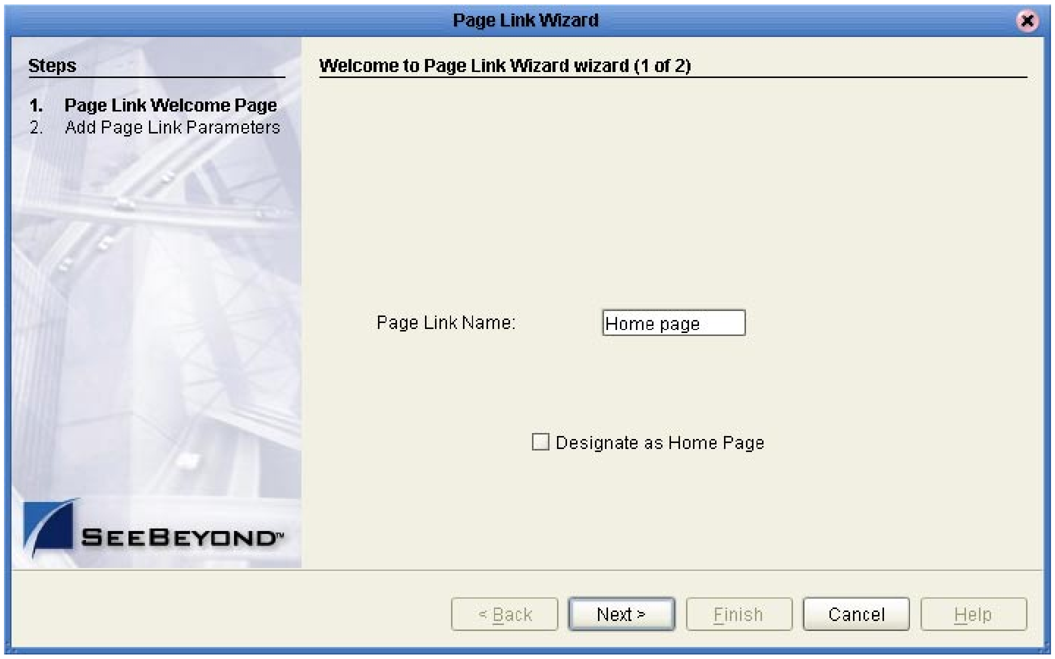
### To create a Page Link

1 In the Project Explorer, right-click the Project.

2 On the context menu, click **New**, and then select **Page Link**.

Step 1 of the Page Link Wizard appears. See Figure 42.

**Figure 42** Page Link Wizard - Page 1



Welcome to Page Link Wizard wizard (1 of 2)

Page Link Name: Home page

☐ Designate as Home Page

3 In the **Page Link Name** field, type a descriptive name for the link.

This is the internal application link name, used by the system to identify the link.

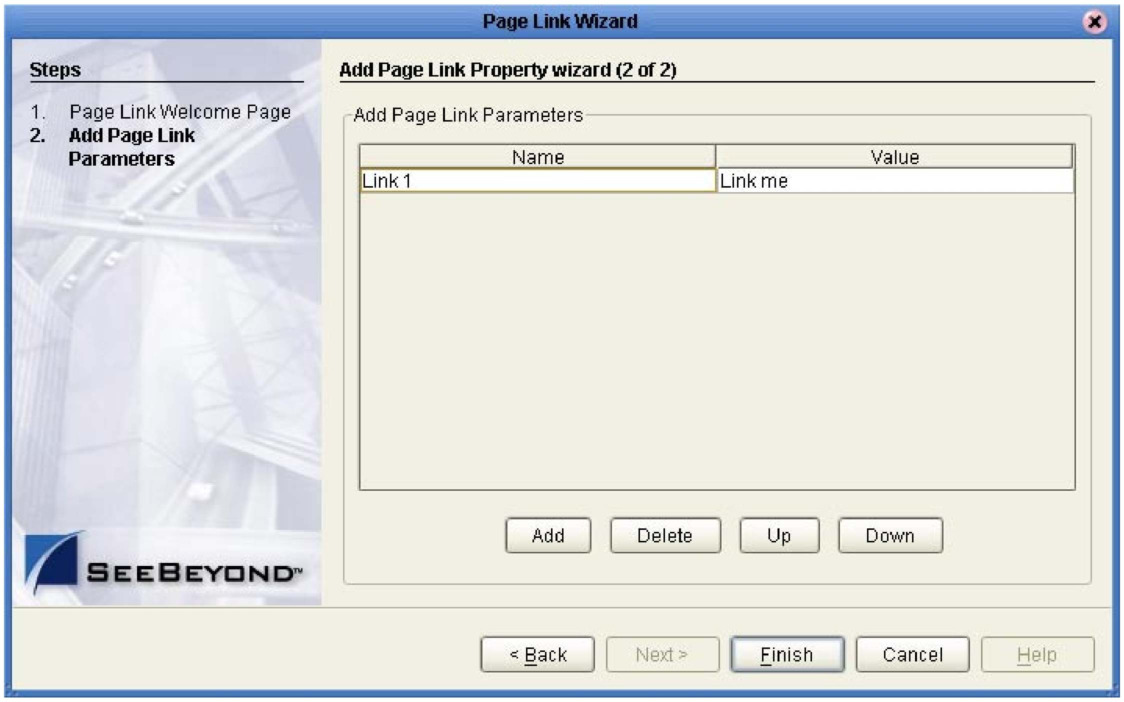
4 If you want the link to be defined as the link to the Page Flow's Home page, check **Designate as Home Page**.

The Home page designation specifies that the page will be the first page that users see when the Web application is accessed via its URL. In a Page Flow, only one page can be designated as the Home page.

5 Click **Next**.

Step 2 of the Page Link Wizard appears. See Figure 43.

**Figure 43** Page Link Wizard - Step 2



| Name | Value |
|---|---|
| Link 1 | Link me |

6 Add Page Link parameters.

A Page Link parameter is any data type that is passed via the link to an HTML or Form Object on the linked destination page. You can pass a single parameter or multiple sets of parameters.

- **Name** is the parameter's name (the internal system name used when specifying a target for the link).
- **Value** is the data to be passed via the link to a target object on the destination page.

**Note:** *A value must be assigned to the page link, even if it is dynamically assigned at runtime. Do not leave the Value field blank.*

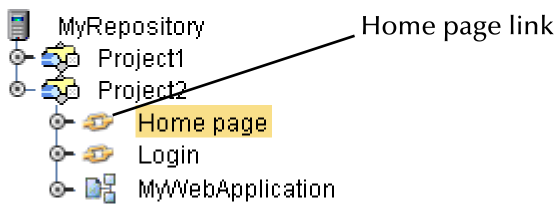
7 Modify the Page Link parameter fields, if necessary:

- To add a Name/Value parameter field to the stack, click **Add**.
- To delete a Name/Value parameter field from the stack, click the field and then click **Delete**.
- To navigate up one level in a stack of parameter fields, click **Up**.
- To navigate down one level, click **Down**.

8 Click **Finish**.

The Page Link is added to your Project in the Project Explorer tree. See Figure 44.

**Figure 44** Links in Enterprise Explorer



You can drag a page link from the Project Explorer tree and drop it directly onto a page in a Page Flow. In a Page Flow, page links can be used as follows:

- To link pages and Activities within the process
- As a Receive operation
- As an Activity in an Event-Based Decision

For information on using Page Links in a Page Flow, see Chapter 7, **“Page Flow Designer Tutorial”**.

Chapter 5

# Page Layout Designer Tutorial

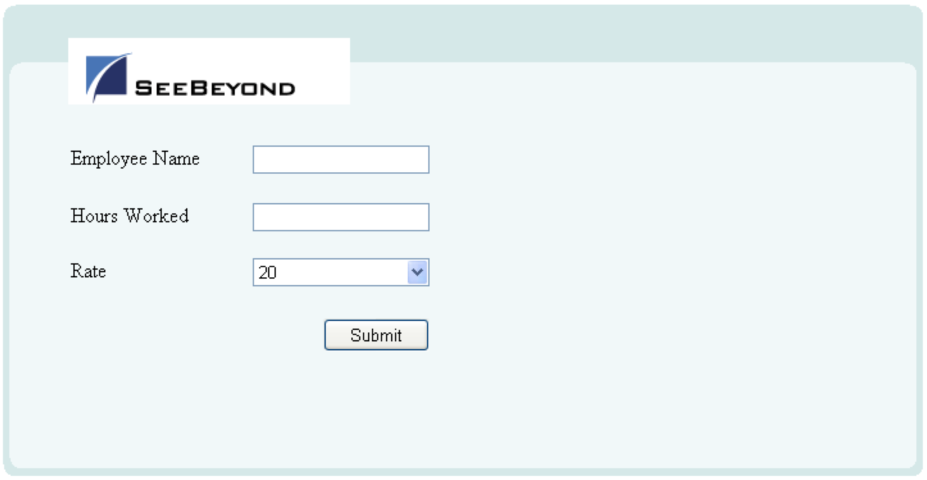
This chapter guides you through the process of starting a new page, placing GUI components on the canvas, and using the property sheets.

**This Chapter Includes:**

## 5.1 Overview

In the following tutorial, you build an input page for a small payroll calculation system. The purpose of the exercise is to create a finished Web page that you can preview in your browser. In the process of constructing the page, you will become familiar with the Page Layout Designer. For an example of the finished page, see Figure 45.

**Figure 45** Finished Web Page



## 5.2 Downloading the Sample Files

The sample files are provided in **.zip** file format. They are included in the eVision Studio documentation **.sar** file. The first step in this tutorial is to download the sample files from the Repository.

The Repository must be running before you can use Enterprise Manager or Enterprise Designer. The first step of any procedure is to make sure that the Repository is running. The *SeeBeyond ICAN Suite Installation Guide* describes how to start the Repository.

### To download the sample files

1. Start Internet Explorer.
2. In the **Address** field, enter **http://*hostname*:*portnumber***

   where

   ***hostname*** is the TCP/IP host name of the server where the Repository is installed.

   ***portnumber*** is the base port number of the Repository.

   The **SeeBeyond Customer Login** window of Enterprise Manager appears.
3. Enter your username and password.
4. Click **Login**.

   The Enterprise Manager home page appears.
5. Click the **DOCUMENTATION** tab.
6. In the left frame of the Documentation page, click **eVision Studio**.

***Note:*** *If you do not see an eVision Studio link in the left frame, then the eVision Studio Documentation **.sar** file must be uploaded. See* **Chapter 3**, **"Installing eVision Studio" on page 25**.

7 In the right frame of the Documentation page, click **Download Sample**.

8 Using an archive utility (such as WinZip), extract the sample files to a folder on your computer. Remember where you saved the files. If you use the default folder names, the sample files will be stored in a directory called **eVision_Sample**.

The sample files are:

- **blueBG.gif**
- **i_seebeyondlogo.gif**
- **eVisionSampleComponents.zip**
- **eVisionTutorial_sample.zip**

In this tutorial, you use the **blueBG.gif** and **i_seebeyondlogo.gif** files.

## 5.3 Starting a New Page Layout

### To start a new Page Layout

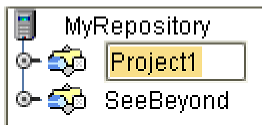
1 Start Enterprise Designer.

2 In the Project Explorer, right-click the Repository.

3 On the context menu, select **New Project**.

A Project icon is added to the Enterprise Explorer tree. The new branch is called **Project*n***, where ***n*** is the next sequential default Project number. The keyboard focus is inside the **Project*n*** field. See Figure 46.

**Figure 46** New Project in Project Explorer



4 Without moving the cursor, type a name for the new Project (for example, **MyProject**).

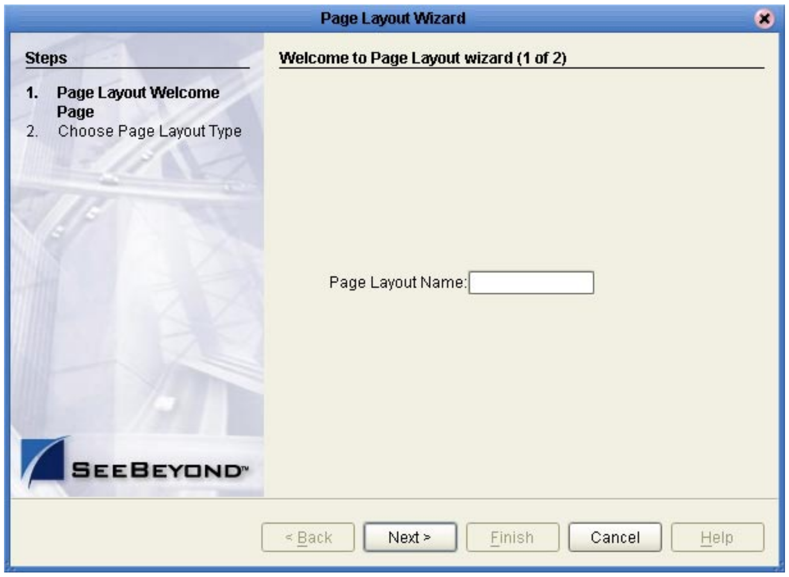
5 Press **Enter**.

6 Right-click the Project.

7 On the context menu, click **New**, and then select **Page Layout**.

Step 1 of the Page Layout Wizard appears. See Figure 47.

**Figure 47** Page Layout Wizard - Step 1



Welcome to Page Layout wizard (1 of 2)

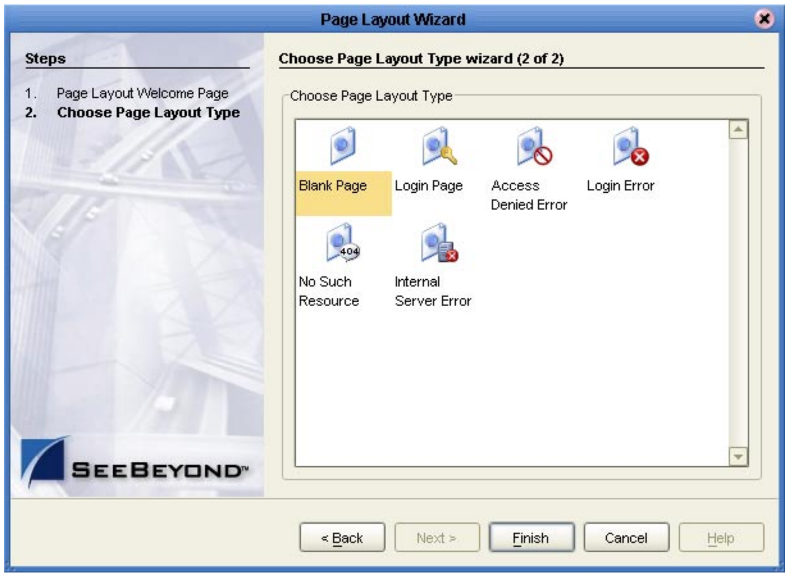
Page Layout Name:

8 In the **Page Layout Name** field, type a unique name for the new Page Layout (for example, **MyWebPage**).

9 Click **Next**.

Step 2 of the Page Layout Wizard appears. See Figure 48.

**Figure 48** Page Layout Wizard - Step 2



Choose Page Layout Type wizard (2 of 2)

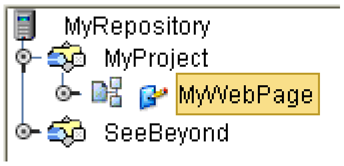
Choose Page Layout Type

10 The wizard allows you to select any of the preconfigured Web pages based on a particular Project's requirements. For this Project, you start with a blank page. Therefore, click the **Blank Page** icon.

11 Click **Finish**.

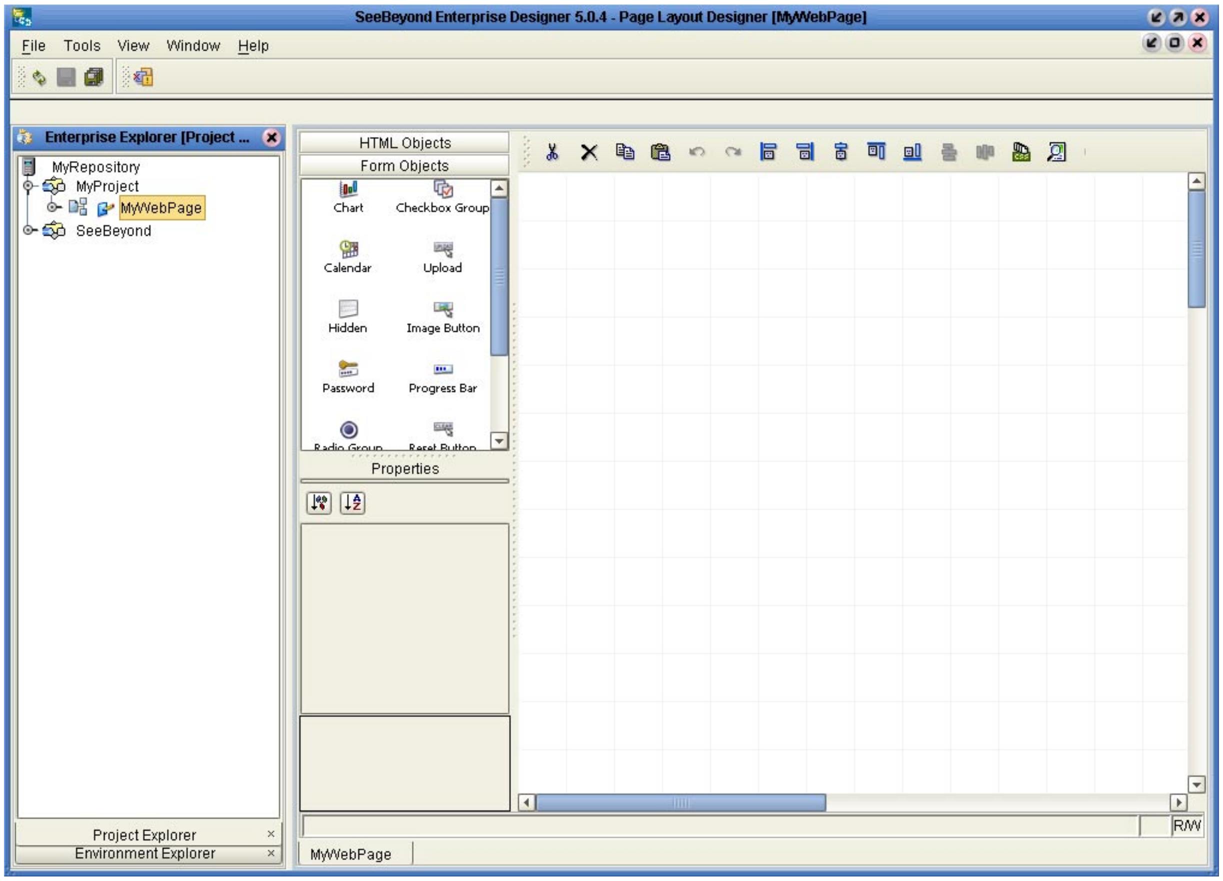
A new Page Layout is added to the Project Explorer. See Figure 49.

**Figure 49** New Page Layout in Project Explorer



The Page Layout Designer appears in the work area of Enterprise Designer with a blank canvas. See Figure 50.

**Figure 50** Page Layout Designer with a Blank Canvas



You have just created a blank Web page and started the Page Layout Designer. You will now use the Page Layout Designer tools and pre-built GUI components to create a Web page that receives user input.

## 5.4 Importing the Image Files

The sample Web page requires that you import two image files for use in the Page Layout: a background image (**blueBG.gif**) and a SeeBeyond logo (**i_seebeyondlogo.gif**). Before graphic images can be used in an eVision Web page, you must import them into the Project.

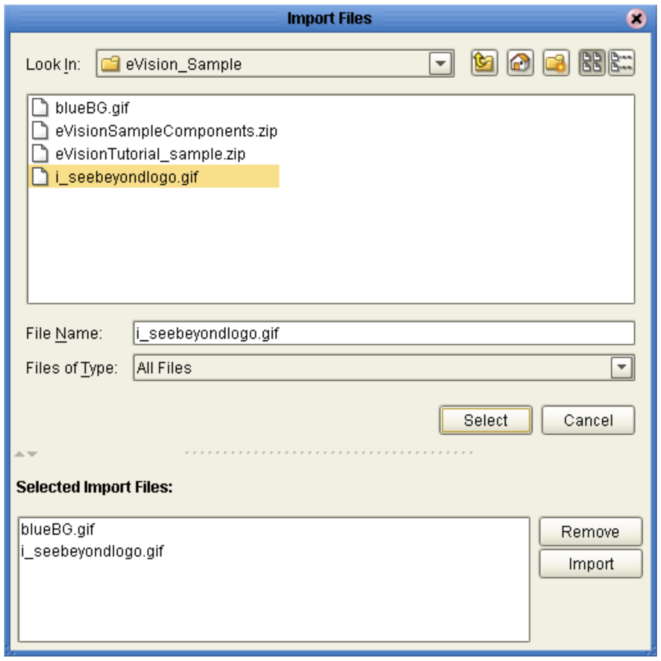
### To import the image files

1. In the Project Explorer, right-click the Project.
2. On the context menu, click **New**, and then select **File**.

   The **Import Files** dialog box appears. See Figure 51.
3. Navigate to the folder containing the image files that you extracted in **"Downloading the Sample Files" on page 70**.

   Graphic images can reside in any directory that is accessible to your computer, locally or on a network.
4. Select **blueBG.gif** and click **Select**. The file name appears in the **Selected Import Files** box.
5. Select **i_seebeyondlogo.gif** and click **Select**. The file name appears in the **Selected Import Files** box.

**Figure 51** Selecting the Image Files



Look In: eVision_Sample

blueBG.gif
eVisionSampleComponents.zip
eVisionTutorial_sample.zip
i_seebeyondlogo.gif

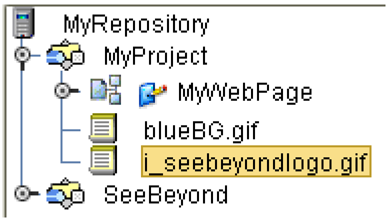
File Name: i_seebeyondlogo.gif
Files of Type: All Files

Selected Import Files:
blueBG.gif
i_seebeyondlogo.gif

6. Click **Import**.

The image files are now part of your Project. See Figure 52.

**Figure 52** Image Files in Project Explorer

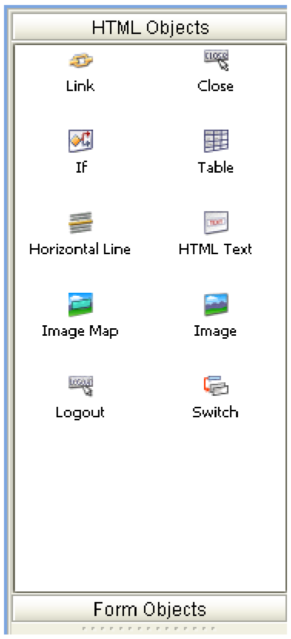


## 5.5 Placing GUI Components on the Canvas

In this section, you place a variety of GUI components on the Page Layout Designer canvas. You also specify some of the components’ properties.

Before you begin, make sure that the **HTML Objects** palette is in front. See Figure 53.

**Figure 53** HTML Objects Palette



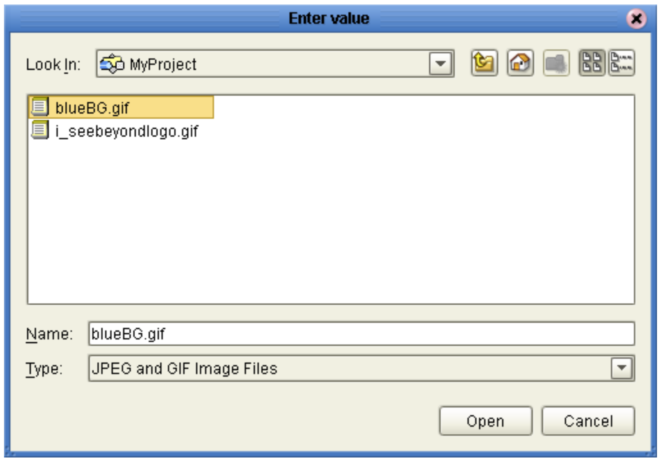
### 5.5.1 Create the Background Layer Component

The **blueBG.gif** file will be used as the background of the Web page.

1 From the **HTML Objects** palette, drag the **Image** icon onto the approximate center of the canvas.

The **Enter value** dialog box appears. See Figure 54.

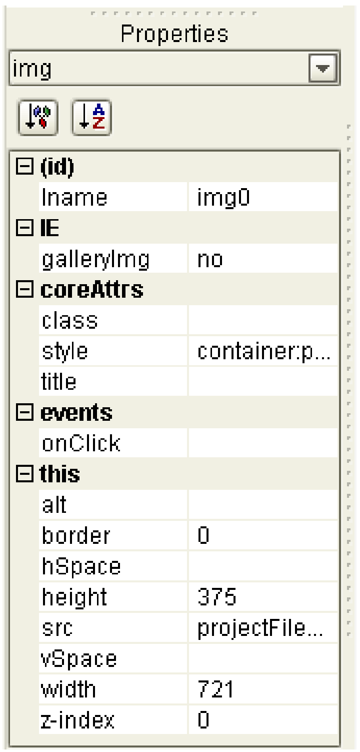
**Figure 54** Enter value Dialog Box



2 Select the **blueBG.gif** file and click **Open**.

The **Image** component appears with a gold outline. The **Properties** sheet for the **Image** component is automatically displayed. See Figure 55.

**Figure 55** Properties Sheet for the Image Component



Properties

img

| | |
|---|---|
| ⊟ **(id)** | |
| lname | img0 |
| ⊟ **IE** | |
| galleryImg | no |
| ⊟ **coreAttrs** | |
| class | |
| style | container:p... |
| title | |
| ⊟ **events** | |
| onClick | |
| ⊟ **this** | |
| alt | |
| border | 0 |
| hSpace | |
| height | 375 |
| src | projectFile... |
| vSpace | |
| width | 721 |
| z-index | 0 |

3 In the **Properties** sheet, define the value for **lname** by doing the following:

A In the left column, select the **lname** property.

B In the right column, delete the default value and enter **Background**.

**lname** is the internal, logical system name for the component, used to identify the component in a Page Flow.

4 In the **Properties** sheet, define the value for **z-index** by doing the following:

A In the left column, select the **z-index** property.

B In the right column, delete the default value and enter **-1**.

**z-index** specifies the placement of the object in the stack order. Setting the **z-index** property to -1 enables you to place components on top of the background layer and have the components be immediately visible. For more information, see **"Moving Overlapping Components in the Z-Direction" on page 54**.

5 If you want to move the component, click the component inside the boundary and drag it to the new location.

### 5.5.2 Create the Page Banner Component

The **i_seebeyondlogo.gif** file will be used as the company logo.

1 From the **HTML Objects** palette, drag the **Image** icon onto the canvas and inside the boundaries of the background image.

The **Enter value** dialog box appears.

2 Select the **i_seebeyondlogo.gif** file and click **Open**.

The **Properties** sheet for the **Image** component is automatically displayed.

3 In the **Properties** sheet, define the value for **lname** by doing the following:

A In the left column, select the **lname** property.

B In the right column, delete the default value and enter **logo**.

4 If you want to move the component, click the component inside the boundary and drag it to the new location. **Figure 45 on page 70** shows the recommended location.

### 5.5.3 Create the Employee Name Label Component

This component identifies the input field where the user enters his or her name.

1 From the **HTML Objects** palette, drag the **HTML Text** icon onto the canvas.

2 Place the component left-aligned with the **logo** component.

The **Properties** sheet for the **HTML Text** component is automatically displayed.

3 In the **Properties** sheet, define the value for **lname** by doing the following:

A In the left column, select the **lname** property.

B In the right column, delete the default value and enter **EmpName**.

4 In the **Properties** sheet, define the value for **text** by doing the following:

A In the left column, select the **text** property.

B In the right column, click the **[default text]** field. The **Enter value for "text"** window appears.

C Delete the **[default text]** string and type **Employee Name**.

D Click **OK**.

5 On the canvas, the text string may appear truncated. To see the entire text string:

A Select the component.

B Press and hold the Shift key.

C Press the right arrow key until the entire text string is visible.

### 5.5.4 Create the Hours Worked Label Component

This component identifies the input field where the user enters the number of hours worked.

1 From the **HTML Objects** palette, drag the **HTML Text** icon onto the canvas.

2 Place the component under and left-aligned with the **EmpName** component.

The **Properties** sheet for the **HTML Text** component is automatically displayed.

3 In the **Properties** sheet, define the value for **lname** by doing the following:

A In the left column, click the **lname** property.

B In the right column, delete the default value and enter **HoursWorked**.

4 In the **Properties** sheet, define the value for **text** by doing the following:

A In the left column, select the **text** property.

B In the right column, click the **[default text]** field. The **Enter value for "text"** window appears.

C Delete the **[default text]** string and type **Hours Worked**.

D Click **OK**.

5 On the canvas, the text string may appear truncated. To see the entire text string:

A Select the component.

B Press and hold the Shift key.

C Press the right arrow key until the entire text string is visible.

### 5.5.5 Create the Rate Label Component

This component identifies the drop-down list where the user selects his or her hourly pay rate.

1 From the **HTML Objects** palette, drag the **HTML Text** icon onto the canvas.

2 Place the component under and left-aligned with the **HoursWorked** component.

The **Properties** sheet for the **HTML Text** component is automatically displayed.

3 In the **Properties** sheet, define the value for **lname** by doing the following:

   A In the left column, click the **lname** property.

   B In the right column, delete the default value and enter **Rate**.

4 In the **Properties** sheet, define the value for **text** by doing the following:

   A In the left column, select the **text** property.

   B In the right column, click the **[default text]** field. The **Enter value for "text"** window appears.

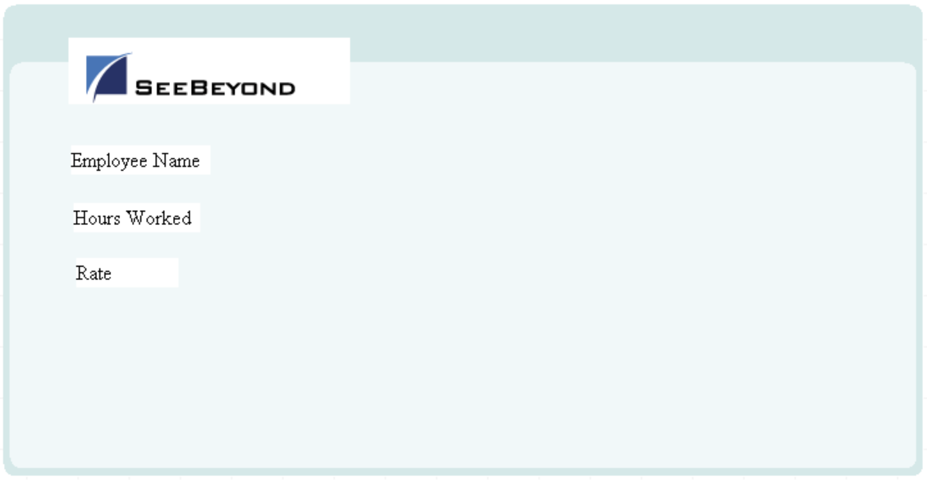
   C Delete the **[default text]** string and type **Rate**.

   D Click **OK**.

5 On the **File** menu, click **Save All**.

Check your progress. The page should look like the example in Figure 56.

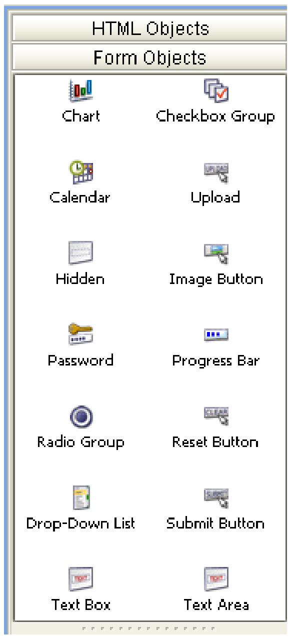
**Figure 56** Progress Check 1



## 5.5.6 Swap the HTML Objects and Form Objects Palettes

To bring the **Form Objects** palette to the front, click the **Form Objects** title bar. See Figure 57.

**Figure 57** Form Objects Palette



## 5.5.7 Create the Employee Name Input Field Component

This component will be displayed to users as an empty input field to the right of the **Employee Name** label.

1. From the **Form Objects** palette, drag the **Text Box** icon onto the canvas.
2. Align the component horizontally with and to the right of the **EmpName** component. Make sure that the components do not overlap.

   The **Properties** sheet for the **Text Box** component is automatically displayed.
3. In the **Properties** sheet, define the value for **lname** by doing the following:
   A. In the left column, click the **lname** property.
   B. In the right column, delete the default value and enter **EmpNameInput**.
4. In the **Properties** sheet, define the value for **value** by doing the following:
   A. In the left column, select the **value** property.
   B. In the right column, delete the **[default text]** string. (You want this input field to be initially blank, to receive input from users.)

### 5.5.8 Create the Hours Worked Input Field Component

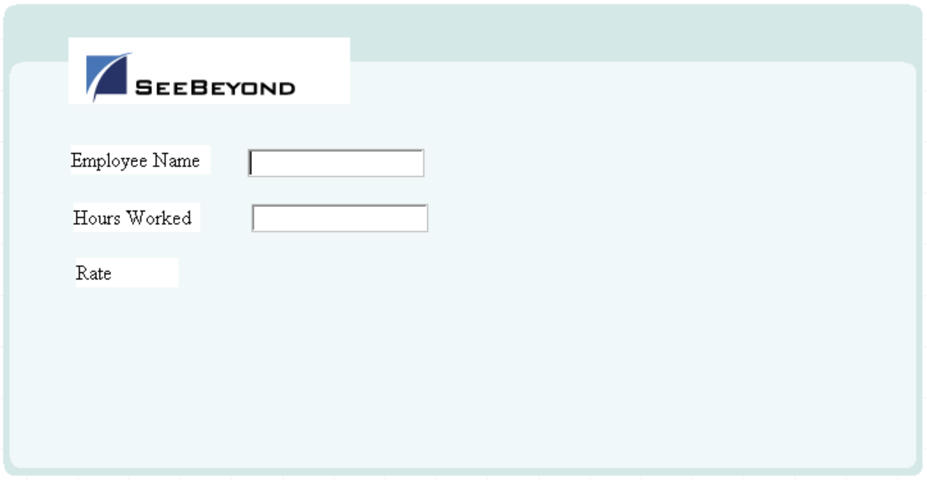
This component will be displayed to users as an empty input field to the right of the **Hours Worked** label.

1. From the **Form Objects** palette, drag the **Text Box** icon onto the canvas.
2. Align the component horizontally with and to the right of the **HoursWorked** component. Make sure that the components do not overlap.

   The **Properties** sheet for the **Text Box** component is automatically displayed.
3. In the **Properties** sheet, define the value for **lname** by doing the following:
   - A In the left column, click the **lname** property.
   - B In the right column, delete the default value and enter **HoursWorkedInput**.
4. In the **Properties** sheet, define the value for **value** by doing the following:
   - A In the left column, select the **value** property.
   - B In the right column, delete the **[default text]** string. (You want this input field to be initially blank, to receive input from users.)
5. On the **File** menu, click **Save All**.

Check your progress. The page should look like the example in Figure 58.

**Figure 58** Progress Check 2



### 5.5.9 Create the Rate Drop-Down List Component

This component will be displayed to users as a drop-down list to the right of the **Rate** label.

1. From the **Form Objects** palette, drag the **Drop-Down List** icon onto the canvas.
2. Align the component horizontally with and to the right of the **Rate** component. Make sure that the components do not overlap.

The **Properties** sheet for the **Drop-Down List** component is automatically displayed.

3 In the **Properties** sheet, define the value for **lname** by doing the following:

   A In the left column, click the **lname** property.

   B In the right column, delete the default value and enter **RateDropDown**.

4 On the canvas, right-click the **Drop-Down List** component.

5 On the context menu, select **Edit Options**.

   The **Edit Options** dialog box appears.

6 Under the **Labels** heading, click the empty input field and type **20**. This is the default pay rate that will be displayed to users in the drop-down list on the Web page.

7 Under the **Values** heading, click the empty input field and type **20.** This is the label for the drop-down list entry.

8 To add three additional input rows:

   A Click **Add** three times.

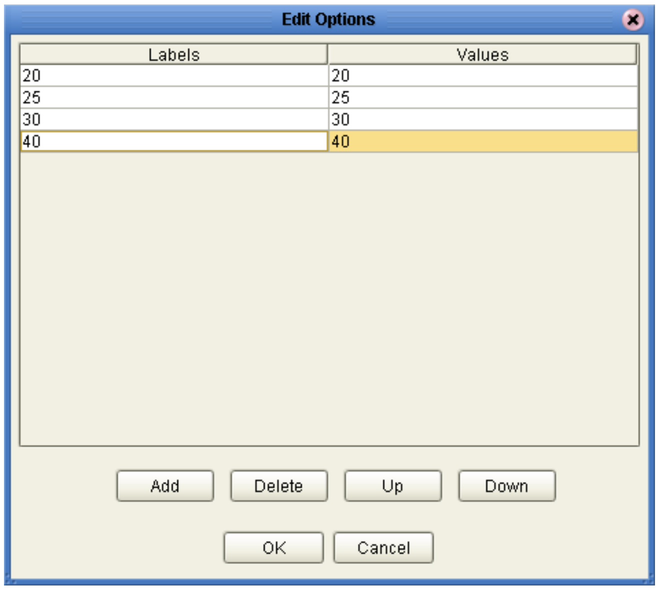
   B In row 2, type **25** under the **Labels** heading and again under the **Values** heading.

   C In row 3, type **30** under the **Labels** heading and again under the **Values** heading.

   D In row 4, type **40** under the **Labels** heading and again under the **Values** heading.

   The **Edit Options** dialog box should now look like Figure 59.

**Figure 59** Edit Options Dialog Box



| Labels | Values |
|---|---|
| 20 | 20 |
| 25 | 25 |
| 30 | 30 |
| 40 | 40 |

9 Click **OK**.

### 5.5.10 Create the Submit Button Component

This component will be displayed to users as a button with the label **Submit**.

1 From the **Form Objects** palette, drag the **Submit Button** icon onto the canvas.

2 Place the component below and right-aligned with the **Drop-Down List** component.

The **Properties** sheet for the **Submit Button** component is automatically displayed.

3 In the **Properties** sheet, define the value for **lname** by doing the following:

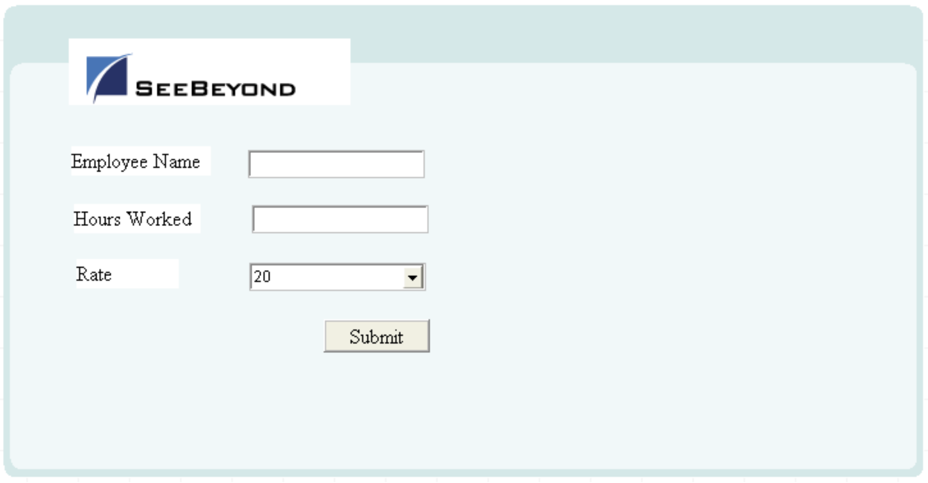
A In the left column, click the **lname** field.

B In the right column, delete the default value and enter **Submit**.

4 On the **File** menu, click **Save All**.

Check your progress. The page should look like the example in Figure 60.

**Figure 60** Progress Check 3



## 5.6 Aligning Objects Using the Alignment Tools

If the GUI components are not precisely aligned, or if the space between the objects is inconsistent, you can adjust the alignment of the components using the alignment tools.

To choose two or more objects to be aligned, do the following:

1 Press and hold the Shift key on your computer.

2 Select the objects that you want to align.

3 Click the desired alignment tool on the Page Layout Designer toolbar. Table 5 describes the alignment tools.

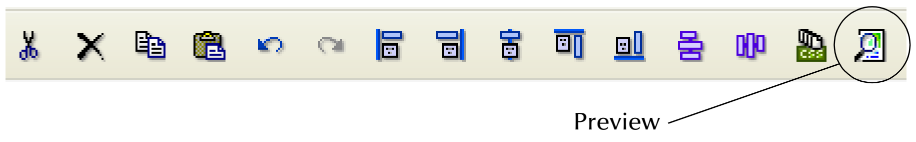
**Table 5** Alignment Tools

| Tool | Name | Purpose |
|---|---|---|
| | Left Align | Aligns two or more selected components with the left-most component in the selected group. |
| | Right Align | Aligns two or more selected components with the right-most component in the selected group. |
| | Center Align | Aligns two or more selected components with the center of the canvas (the center of the Web page). |
| | Top Align | Aligns two or more selected components with the top-most component in the selected group. |
| | Bottom Align | Aligns two or more selected components with the bottom component in the selected group. |
| | Vertical Spacing | Creates equal vertical spacing between object in a group based on an averaging algorithm. |
| | Horizontal Spacing | Creates equal horizontal spacing between objects in a group based on an averaging algorithm. |

## 5.7 Previewing the Finished Web Page

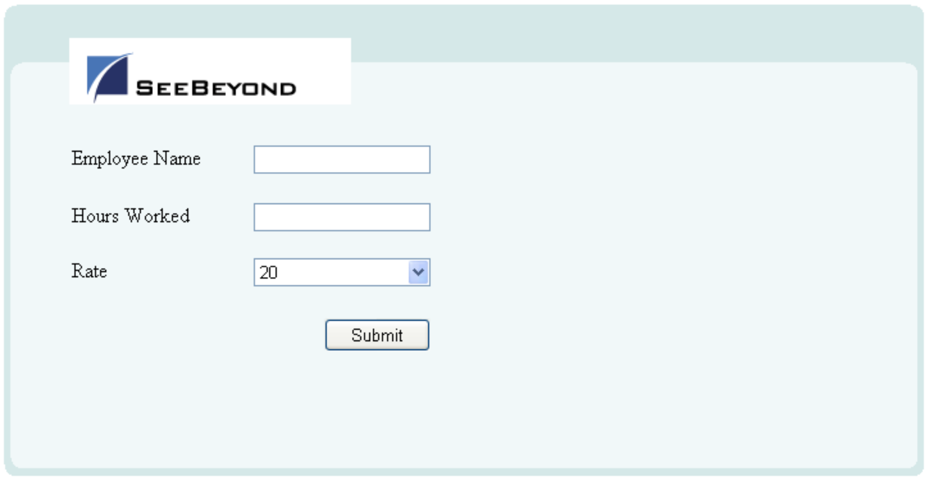
To preview the finished Web page in your browser, click the **Preview** icon on the Page Layout Designer toolbar.

**Figure 61** Page Layout Designer Toolbar



The page should look like the example in Figure 62.

**Figure 62** Finished Web Page in Preview Mode



You can enter text in the text boxes and select values from the drop-down list. However, the submit button does not work.

## 5.8 Using the Version Control System

The Version Control system allows you to maintain multiple versions of selected Project or Environment components. The version history of each component is recorded to a log file, and can be viewed by means of a menu option. For more information, see the *eGate Integrator User's Guide*.

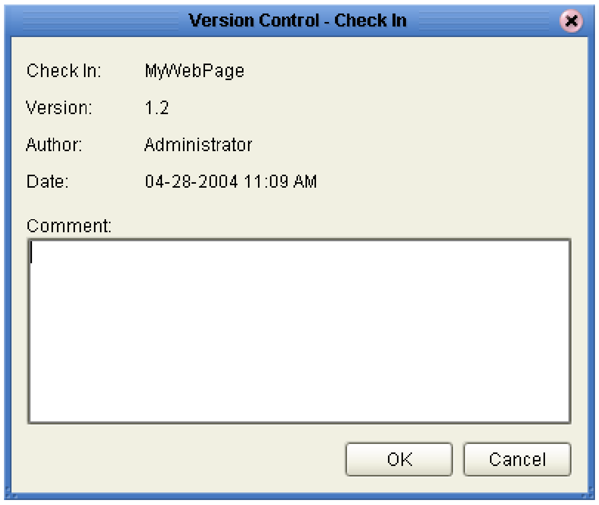
You can practice using the Version Control system with the Page Layout that you created in this chapter.

**To check in a Page Layout**

1. In the Project Explorer, right-click the Page Layout.
2. On the context menu, select **Check In**.

   The **Version Control - Check In** dialog box appears. See Figure 63.

**Figure 63** Version Control - Check In Dialog Box



Check In: MyWebPage
Version: 1.2
Author: Administrator
Date: 04-28-2004 11:09 AM
Comment:

3 In the **Comment** box, enter a comment.

4 Click **OK**.

The writing pad icon to the right of the Page Layout icon disappears. A red lock in the lower-left corner of the Page Layout icon appears.

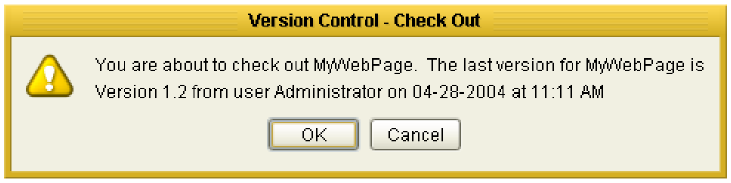
**To check out a Page Layout**

1 In the Project Explorer, right-click the Page Layout.

2 On the context menu, select **Check Out**.

The **Version Control - Check Out** dialog box appears. See Figure 64.

**Figure 64** Version Control - Check Out Dialog Box



You are about to check out MyWebPage. The last version for MyWebPage is Version 1.2 from user Administrator on 04-28-2004 at 11:11 AM

3 Click **OK**.

The red lock in the lower-left corner of the Page Layout icon disappears. A writing pad icon to the right of the Page Layout icon appears.

Chapter 6

# Using the Page Flow Designer

To build an eVision Studio Web application, you first create the individual user-facing pages that you want users to interact with. You then use the Page Flow Designer to link the Web pages together and create a logical chain called a Page Flow.

**This Chapter Includes:**

## 6.1 Creating a Page Flow

A Page Flow is a structured series of Web pages that comprise a Web-enabled business process.

A Web-enabled business process can be an internal service, or it can be exposed as an external application over the Web. A Page Flow can involve a variety of participants, and may include internal and external computer systems. When you create a Page Flow, you are creating a graphical representation of what will become a fully functional and deployable Web application.

### 6.1.1. Adding a Page Flow to a Project

When you add a Page Flow to a Project, the Page Flow Designer opens an empty Page Flow canvas that enables you to place pages and other design elements and then connect them together in a sequence. Before you can start building a Page Flow, you must first add the Page Flow to your Project.

### To add a Page Flow to a Project

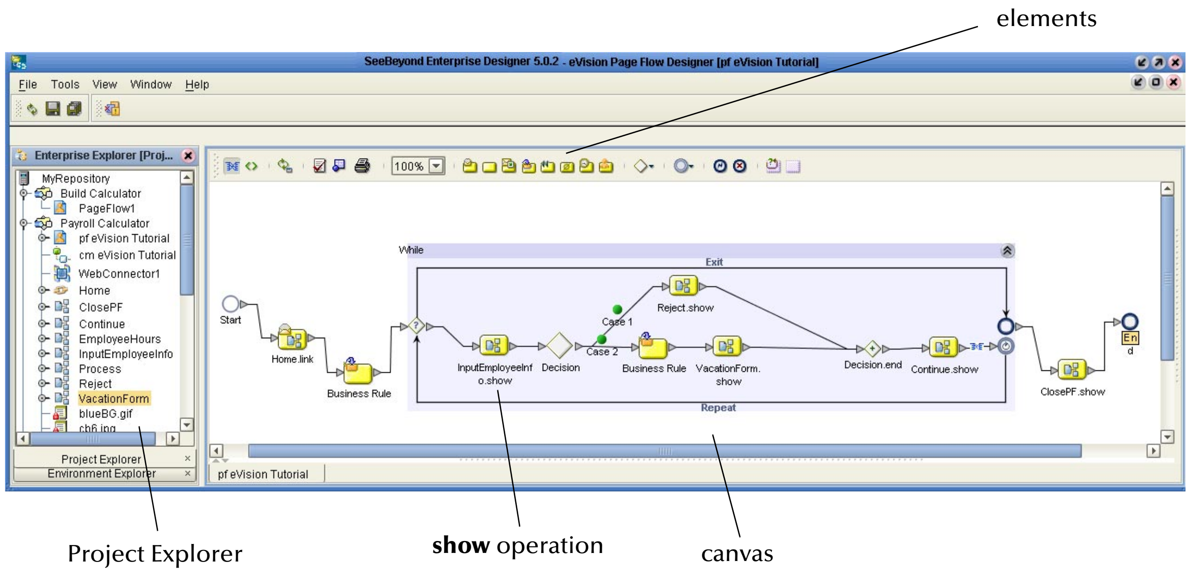
1. In the Project Explorer, right-click the Project.
2. On the context menu, click **New**, and then select **Page Flow**.
3. If desired, change the default name of the Page Flow.

## 6.1.2. Adding Components to a Page Flow

After you add a Page Flow to a Project, you drag and drop Page Flow elements and Web page operations onto the canvas. An example of a Web page operation is the show **operation** of a Page Layout. You then link the components together.

The Page Flow Designer provides the tools to lay out and connect Page Flow elements and other specialized elements. Figure 65 shows an example of the Page Flow Designer GUI.

**Figure 65** Page Flow in the Page Flow Designer



When you start a new Web application project, the **Start** and **End** icons automatically appear on the blank Page Flow Designer canvas. There can be only one starting point for a Page Flow. There can be multiple end points.

### To add components to a Page Flow

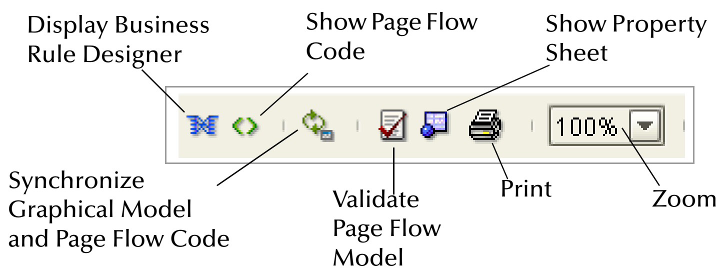
1. Drag the desired Page Flow elements and Web page operations onto the Page Flow Designer canvas.
2. Create links between the elements.
3. Save the Page Flow to the Repository.

   This action validates the connectivity of the Page Flow, generates the code to run it, and saves the Page Flow in the Repository.

## 6.2 Page Flow Designer Tools

The Page Flow Designer tools are shown in Figure 66.

**Figure 66** Page Flow Designer Tools


Display Business Rule Designer
Show Page Flow Code
Show Property Sheet
100%
Synchronize Graphical Model and Page Flow Code
Validate Page Flow Model
Print
Zoom

The Page Flow Designer tools are described in Table 6.

**Table 6** Page Flow Designer Tools

| Tool | Name | Description |
|---|---|---|
| ⋈ | **Display Business Rule Designer** | Opens a window in the lower portion of the Page Flow Designer that enables you to configure relationships between Input and Output attributes. |
| <> | **Show Page Flow Code** | Opens a window in the lower portion of the Page Flow Designer and displays the code generated by the Page Flow Designer. |
| ⟳ | **Synchronize Graphical Model and Page Flow Code** | Synchronizes the Page Flow on the canvas with the underlying code generation. |
| ☑ | **Validate Page Flow Model** | Checks for and reports on execution errors in the Page Flow code. |
| ▦ | **Show Property Sheet** | Allows you to set alert and logging properties for the Page Flow. |
| 🖨 | **Print** | Prints the Page Flow screen image. Allows you to control the scale of the printed image. |
| 100% ▾ | **Zoom** | Increases or decreases the Page Flow image on the screen. |

## 6.3 Page Flow Elements

The Page Flow Designer provides elements that allow you to customize and extend a Page Flow. Pages are dragged from the Enterprise Explorer and dropped onto the design canvas. In addition to Web pages, Page Flows can consist of combinations of basic elements, branching elements, and intermediate events.

Chapter 7, **"Page Flow Designer Tutorial"** illustrates how to use many of these elements.

### 6.3.1. Basic Elements

You can add several types of basic elements to a Page Flow. In addition, the Start Node and End Node elements are automatically added to a Page Flow. See Table 7.

**Table 7** Basic Elements

| Button | Command | Function |
|---|---|---|
| ○ | Start Node | The **Start Node** element indicates the start of the Page Flow. The Start Node is automatically added when you create a Page Flow. A Start Node can connect to a **Page Link** or an **Event Based Decision** element. |
| ▷—▷ | Link | **Links** define the connectivity of the Page Flow by connecting page and sub-process elements together.<br><br>When you select a link, a context menu allows you to configure how data is going to be passed to and from the underlying component or Web Service using Attributes. The Page Flow Designer ensures that the Page Flow is correctly linked by rejecting invalid links.<br><br>Links can also accept mapped values. A link with mapped values will displayed with the "map" icon. |
| ● | End Node | The **End Node** element indicates the completed state of a Page Flow. The End Node is automatically added when you create a Page Flow. |
| ▭ | Receive | The **Receive** element can connect to a **Page Link** or can connect to a **Start** node via a **Page Link**. It is used to indicate the invocation of the Page Flow. The Receive element represents the actual method by which a Page Flow is initiated, for example: A user types a URL into the browser and a servlet initiates the Page Flow. |
| ▭ | Activity | An **Activity** is a step in the Page Flow in which the Page Flow Engine will invoke a Web service or an eGate component. Depending on the configuration of the component, a response may or may not be required. One example would be a synchronous extraction process from a database to return the credit status of a trading partner. |

| Button | Command | Function |
|---|---|---|
|  | Reply | The **Reply** element allows a Page Flow to respond to the external system or user that originally invoked the Page Flow. The original Receive at the beginning of the business process is paired with the Reply at the end of the process. In cases where a message must be sent back to the caller of the process, the Reply uses information that correlates the message in the calling system.<br><br>A Reply acts as the last step in a Page Flow, in which the process is acting as a Web service or sub-process. A Reply correlates the outbound message back to the calling process, for example, it can reply to an external system as a Web service. |
|  | Business Rule | The **Business Rule** element sets data values, including task assignments. It is used when pages have multiple data mappings between the invocation of human tasks or automated systems. |
|  | Compensate | The **Compensate** element is used to invoke compensation on an inner scope that has already completed normally. This construct can be invoked only from within a fault handler or another compensation handler. |
|  | Empty | The **Empty** element allows data to pass through without changes. |
|  | Wait | The **Wait** element acts as a timer. If the user builds a Page Flow in which there are two simultaneous paths within a set framework (one for the page flow, one for the timer, if the timer condition takes place first, an exception will be thrown, handled, and the Page Flow will then be abandoned. |

### 6.3.2. Branching Elements

*Branching elements* allow you to specify the logical flow of information. eVision provides three types of branching elements: Decision, Event Based Decision, and Flow. See Table 8.

**Table 8** Branching Elements

| | | |
|---|---|---|
| ◇ | Decision | The **Decision** element allows one of several possible paths to execute, based on expression logic. This element is used to create complex expressions that determine the path of the Page Flow. It also contains the expression and connection names.<br><br>The Decision element allows you to define expressions that are evaluated to determine the routing of the Page Flow. Expressions are built using the mapping interface and Page Flow attributes.<br><br>**Note:** The Decision element is structured to automatically raise a run-time Exception to alert you to Page Flow errors during construction. You can set this value to Return True or Return False. However, it is not recommended during the construction phase; errors can be hidden during development and only surface in the post-deployment, run-time environment. |
| ◇ | Event Based Decision | The **Event Based Decision** element allows one of several possible paths to execute, based on which link the user has selected. |
| ◇ | Flow | The **Flow** element specifies that one or more pages and/or processes are to flow concurrently. |

**To add a Branching element to the Page Flow Designer canvas**

1. On the Page Flow Designer toolbar, click the expansion arrow on the **Branching Activities** icon.
2. Click the Branching element that you want to use and drag it to the Page Flow Designer canvas.

### 6.3.3. Intermediate Events

*Intermediate events* are elements that can interrupt a Page Flow. Some intermediate events handle exceptions that may occur at run-time or compensate for exceptions. See Table 9.

**Table 9** Intermediate Events

| | | |
|---|---|---|
| | Compensation Handler | The **Compensation Handler** is used when something in a Page Flow fails and requires a rollback based on upstream activities. On an automatic basis in the Page Flow, upstream steps in the Page Flow are notified that the failure has occurred and certain transactions need to be reversed, sometimes in a sequential order. The Compensation Handler allows you to design the process and circumstances in which the compensation takes place. |
| | Catch Named Exception | Each automated system (backend system) or Web service can publish their possible error codes (for instance, fault 15 is "bad data"). Those codes can be mapped to exception handlers. Each exception handler is connected to the scope that surrounds one or more steps in a Page Flow. The components within that scope will throw the exceptions when errors occur and the exception handler will automatically initiate the appropriate process to handle the problem. |
| | Catch All Exceptions | The **Catch All Exceptions** handler is configured to handle all exceptions. |
| | Message Event | The **Message Event** is similar to a Receive Activity, but it occurs only in the middle of a process. Each of these elements can be a different message. |
| | Timer Event | The **Timer Event** imposes a time-out condition on pages, groups of pages, or a Page Flow as a whole to ensure that processes complete within a specified time-frame. Conditions also allow the creation of the process that takes place after a time-out condition takes place. |
| | Throw | The **Throw** handler throws exceptions. |
| | Terminate Process | The **Terminate Process** handler ends the Page Flow. |

**To add an Intermediate event to the Page Flow Designer canvas**

1. On the Page Flow Designer toolbar, click the expansion arrow on the **Intermediate Events** icon.
2. Click the Intermediate event that you want to use and drag it to the Page Flow Designer canvas.

## 6.3.4. While

A **While** loop allows you to encapsulate all or part of a Page Flow within a looping process. See Table 10.

**Table 10** While loop

| | | |
|---|---|---|
| | While | The **While** loop creates and maintains a looping process within a Page Flow. A loop continues a process until an event takes place that signals that the Page Flow is to continue. |

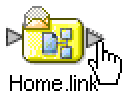
## 6.3.5. Links

In a Page Flow, you connect Page Flow elements manually using links. Links are used to create the flow between Pages.

### To link Web pages and other Page Flow elements

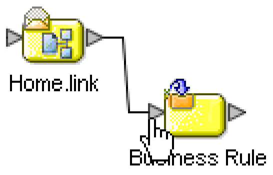
1. Move your cursor over the connector portion of the Page Flow element.
2. Hold the cursor over the outside edge of the element until it changes from the arrow pointer to the hand pointer. See Figure 67.

**Figure 67** Starting a Link



3. Drag a line from the forward connector on the first Web page to the connector on the second Web page or design element, then release the mouse. See Figure 68.

**Figure 68** Finished Link



**Note:** *On a page with a mix of links and HTML forms, the links need to correspond to top-level entry points of the Page Flow, or to message events that trigger the Page Flow.*

## 6.3.6. Validating a Page Flow

After creating a Page Flow, you can check to see if there are any problems, such as pages that are not connected or an incorrect number of output links from a page.

To check the Page Flow for errors, click **Validate Page Flow Model** on the Page Flow Designer toolbar. If there are errors, a message box displays information about the errors. If there are no errors, a message appears stating that there were no errors.

### 6.3.7. Saving a Page Flow

Even if a Page Flow is unfinished and/or contains errors, you can save it as a work in progress and return to it later. To save an unfinished Page Flow, do one of the following:

- On the **File** menu, click **Save**.
- Press **Ctrl+S** on your computer keyboard.
- On the Enterprise Designer toolbar, click **Save**.

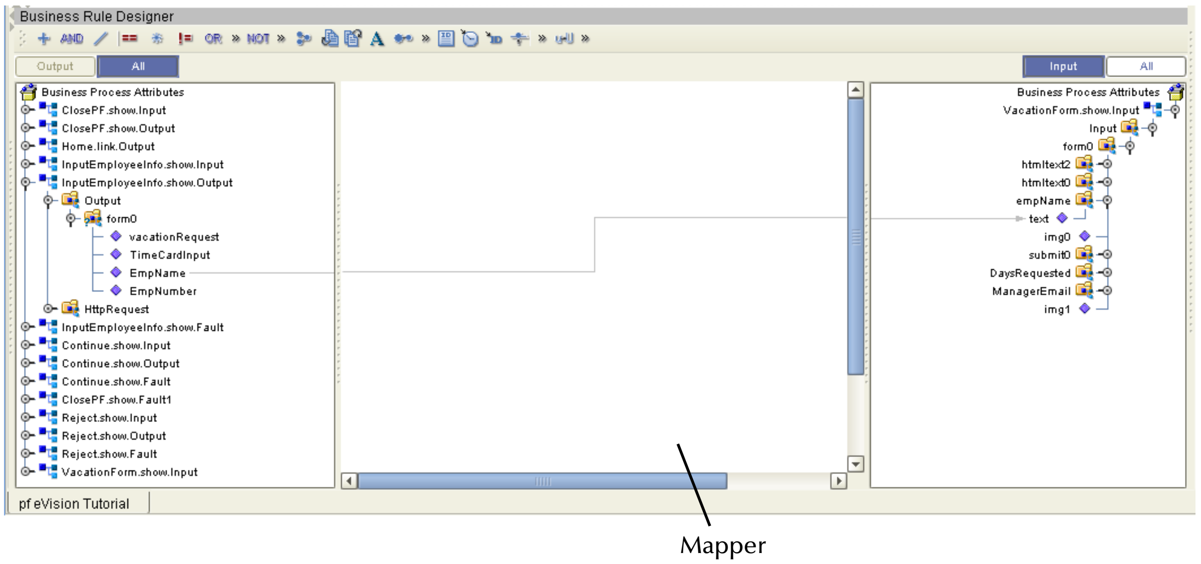
## 6.4 Configuring Page Flow Designer Elements

Some elements in the Page Flow Designer have configurable options. This section describes the elements and explains how to configure the options.

### 6.4.1. About Business Rule Designer

The Business Rule Designer allows you to configure relationships between Input and Output Attributes. Some attributes are automatically configured for each sub-process when you drag and drop a component on the Page Flow Designer. The area where you map attributes in the Business Rule Designer is called the *Mapper*. See Figure 69.

**Figure 69** Business Rule Designer



The Business Rule Designer appears when you click the **Display Business Rule Designer** icon on the Page Flow Designer toolbar, or when you double click an inline Business Rule or Assign element.

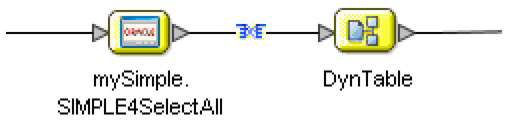
### 6.4.2. Adding an Inline Business Rule

You can add Business Rules to a link that connects two Page Flow elements.

**To add an inline Business Rule**

1. In the Page Flow, right-click a link between two Page Flow elements.
2. On the context menu, select **Add Business Rule**. The Business Rule icon appears on the link.

**Figure 70** Business Rule Icon

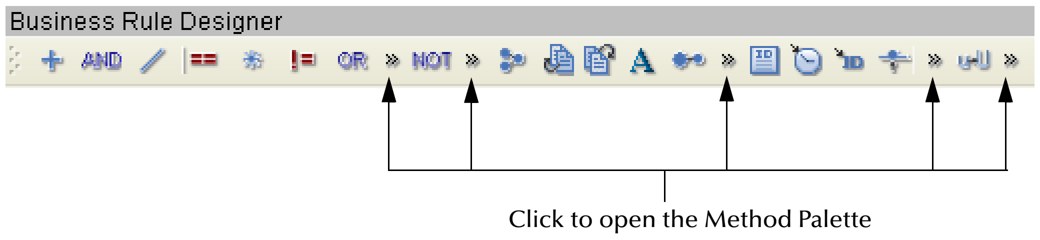


### 6.4.3. Method Palette

Use the Method Palette in the Business Rule Designer to configure data that is passed between input and output pages. You can drag and drop a method from a Method Palette to the Business Rule Designer and then configure the method.

To open the Method Palette, click the horizontal chevrons on the Business Rule Designer toolbar. See Figure 71.

**Figure 71** Business Rule Designer Toolbar



**Appendix A** describes the methods.

## 6.4.4. Invoking Another Page Flow or a Business Process

From within a Page Flow, you can invoke another Page Flow or an eInsight Business Process. The latter scenario assumes that you have installed eInsight Business Process Manager.

This feature lets you create reusable Page Flows and Business Processes.

**To invoke another Page Flow or a Business Process**

1. In the parent Page Flow, add an empty **Activity** element as a placeholder for the child Page Flow or Business Process.
2. In the child Page Flow or Business Process, add an empty **Receive** element at the beginning and a **Reply** element at the end.
3. In the child Page Flow or Business Process, create a WSDL file that will represent the inputs and outputs by doing the following steps:
   - A In the Project Explorer, right click the parent Page Flow.
   - B On the context menu, select **Properties**.
   - C Select the **WSDL** tab.
   - D Click **Create**. The **WSDL Interface Designer** window appears.
   - E Specify values for **PortType**, **Operation**, **Input**, and **Output**.
   - F Click **OK**.
   - G Open the property sheet for the initial **Receive** element and select the appropriate partner, port type, and operation.
   - H Do the same for the **Reply** element.

***Note:*** *If the child Page Flow or Business Process is to be invoked as a synchronous request/reply Web service, then the Receive and Reply elements must have the same partner, port type, and operation.*

4. Save the child Page Flow or Business Process.
5. Return to the parent Page Flow. From the Project Explorer, drag the first operation under the child Page Flow or Business Process to the placeholder that you created in step 1.
6. Perform any appropriate mappings using **Assign Activities** in the parent Page Flow.
7. Deploy the parent Page Flow and the child Page Flow or Business Process.

   For more information on how to add WSDL to a Page Flow, see **“WSDL Files” on page 111**.

### 6.4.5. Dynamic Tables

This section describes how to configure dynamic tables from the Business Rule Designer.

#### Mapping Data into a Dynamic Table

After you create a dynamic table in the Page Layout Designer, you map data into the table in the Page Flow Designer.

Figure 72 shows a dynamic table that will be used as an example. The first row is static. It contains two **HTML Text** components that display column headings. The second row is dynamic. It contains two **Text Box** components.

**Figure 72** Dyanmic Table Example

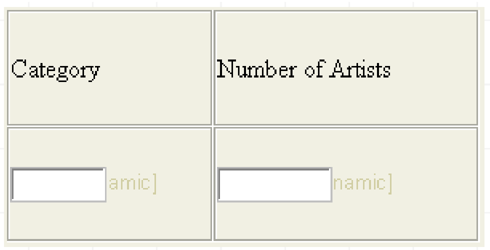
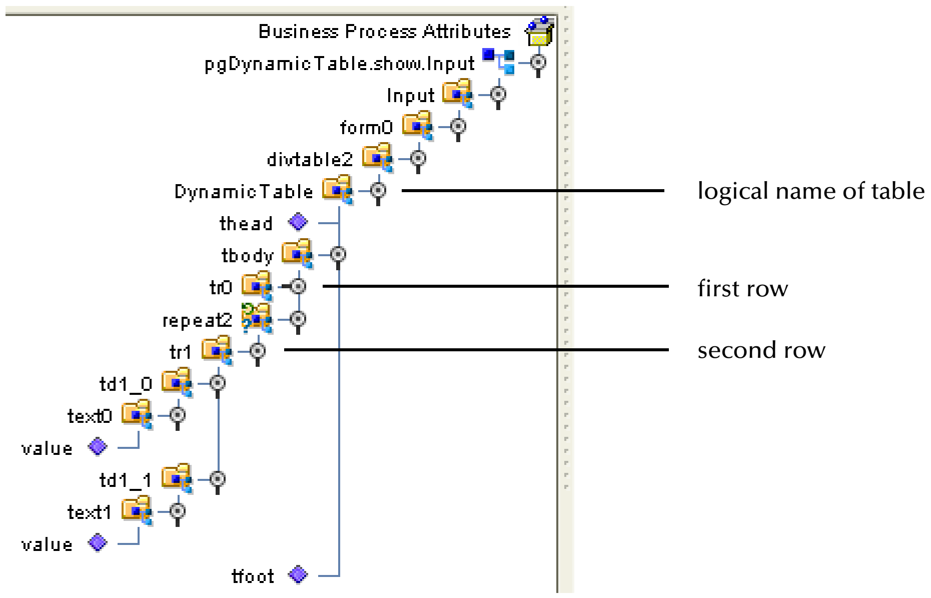


Figure 73 shows how the dynamic table appears in the Business Rule Designer.

The logical name of the table is **DynamicTable**. The **tr0** node represents the first row. The **tr1** node represents the second row. Because the second row is dynamic, it appears within a repeating node (called **repeat2**). Repeating nodes are highlighted with a green, circular arrow.

**Figure 73** Dynamic Table Example in the Business Rule Designer



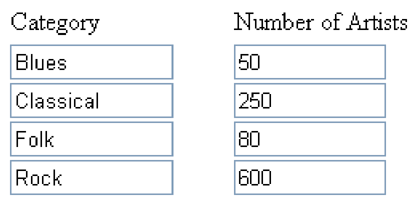
The **td1_0** node represents the first cell in the second row. The **Text Box** component is called **text0**.

The **td1_1** node represents the second cell in the second row. The **Text Box** component is called **text1**.

To populate the dynamic row, you map data from an input source (such as an Oracle database) into the appropriate nodes. In the dynamic table example, the data will be mapped into the **value** node of each **Text Box** component.

At runtime, the repeating node reads data from the input source until no data is left. See Figure 74.

**Figure 74** Dynamic Table Example at Runtime



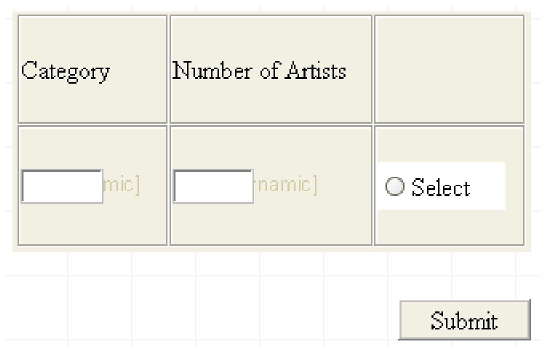
| Category | Number of Artists |
|---|---|
| Blues | 50 |
| Classical | 250 |
| Folk | 80 |
| Rock | 600 |

## Using Predicates in a Dynamic Table

You can select one row from a set of dynamically generated rows by creating a predicate.

Figure 75 shows a modified version of the dynamic table in Figure 72. The table now contains three columns. The third cell in the dynamic row contains a **Radio Group** component. A **Submit Button** component has been added below the table.

**Figure 75** Dynamic Table with Radio Group Component



At runtime, the user chooses one of the dynamically generated rows by selecting the radio button in the third cell. The user then clicks the Submit button.

Typically, you want to do something with the row that the user selected. For example, you could display additional information about the user's selection in a new page. To select the row (rather than all of the dynamically generated rows), you can create a predicate in the Business Rule Designer.

The predicate represents a condition. If the condition is met, the mappings underneath the predicate take place. In the dynamic table example, if the value of the radio button is equal to true, the radio button for this particular row is selected.

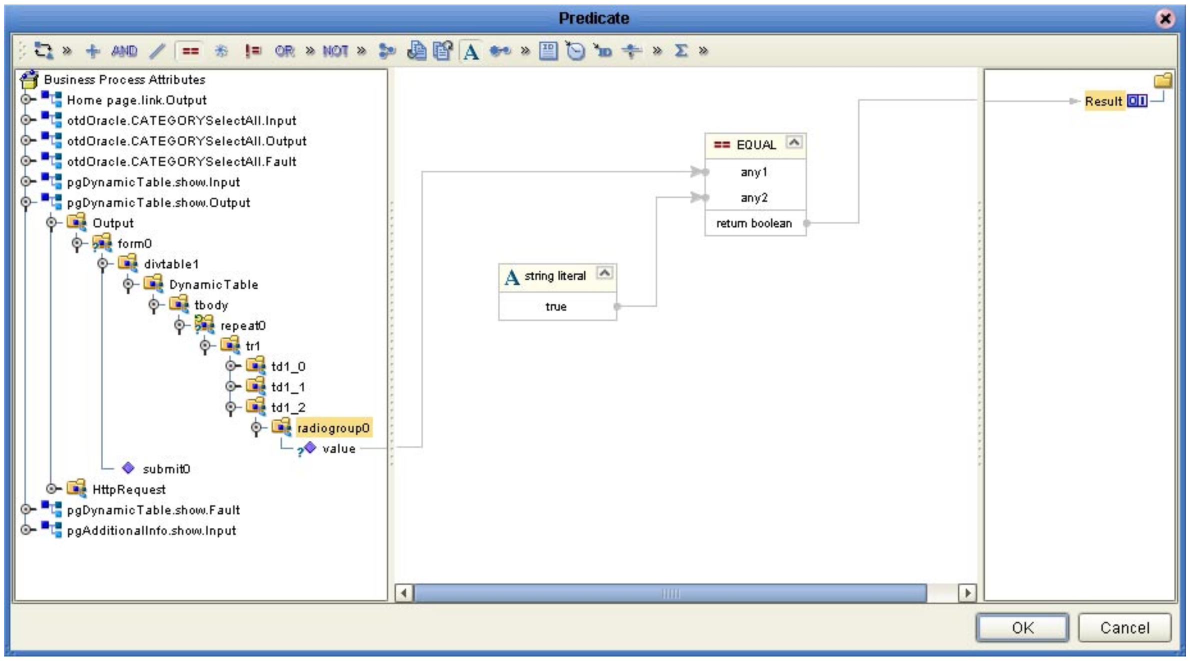
Once you create a predicate, the Business Rule Designer displays the predicate version of the repeating node immediately below the original version. You then perform the appropriate mapping from the predicate version.

You can edit or delete an existing predicate.

### To create a predicate in a dynamic table

1. In the Business Rule Designer, right-click the repeating node that contains the dynamic row.
2. On the context menu, select **New Predicate**. The **Predicate** window appears.
3. Create the condition. The condition in Figure 76 states that the value of the radio button is equal to true.

**Figure 76** Predicate Window


Predicate
AND
OR
NOT
Business Process Attributes
Home page.link.Output
otdOracle.CATEGORYSelectAll.Input
otdOracle.CATEGORYSelectAll.Output
otdOracle.CATEGORYSelectAll.Fault
pgDynamicTable.show.Input
pgDynamicTable.show.Output
Output
form0
divtable1
DynamicTable
tbody
repeat0
tr1
td1_0
td1_1
td1_2
radiogroup0
value
submit0
HttpRequest
pgDynamicTable.show.Fault
pgAdditionalInfo.show.Input
== EQUAL
any1
any2
return boolean
string literal
true
Result
OK
Cancel

4. Click **OK**.

   The Business Rule Designer displays the predicate version of the repeating node immediately below the original version. The syntax of the predicate appears in square brackets.
5. You can now map data from the predicate version of the repeating node.

Figure 77 shows a repeating node that contains a **Radio Group** component. The data is mapped to an **HTML Text** component in a new page.

**Figure 77** Mapping from Predicate Version of Repeating Node (Radio Group)

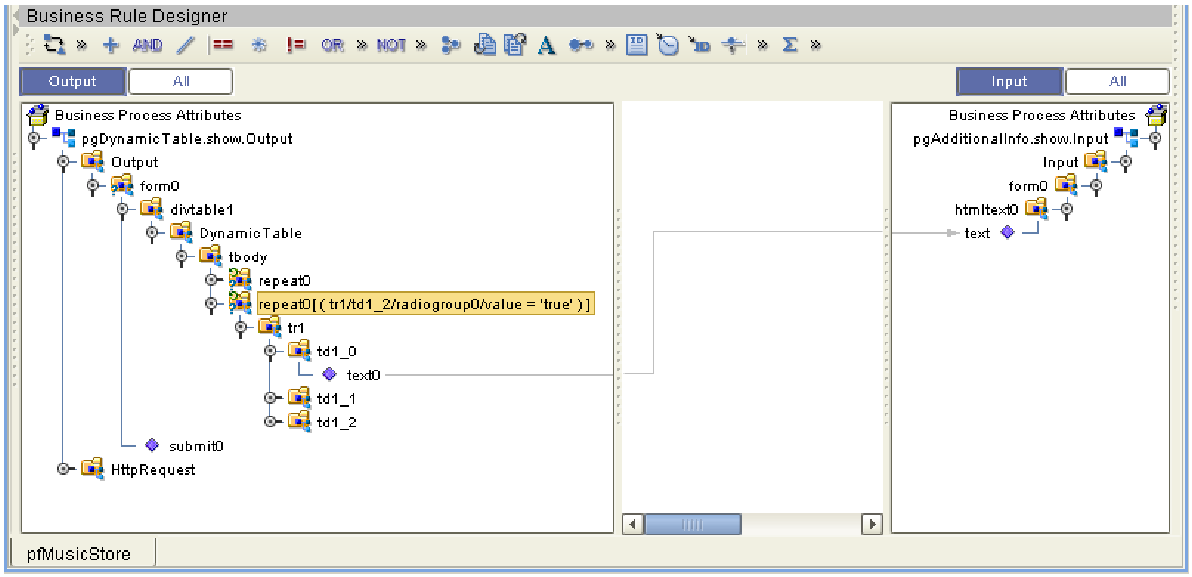
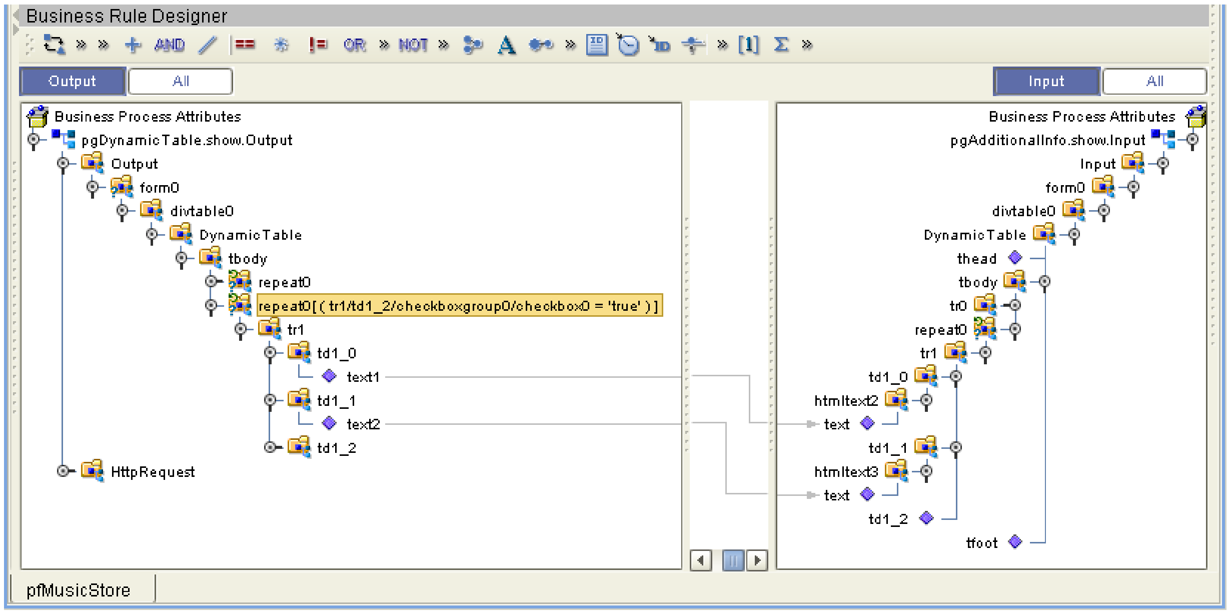


Figure 78 shows a repeating node that contains a **Check Box Group** component. Because the user can select more than one check box at runtime, this scenario differs from the **Radio Group** scenario. The repeating node returns repeating data. The data is mapped to a dynamic table in a new page.

**Figure 78** Mapping from Predicate Version of Repeating Node (Check Box Group)



### To edit a predicate

1. In the Business Rule Designer, right-click the predicate.
2. On the context menu, select **Edit Predicate**.
3. Click **Yes**. The **Predicate** window appears.
4. Make your changes.
5. Click **OK**.

### To delete a predicate

1. In the Business Rule Designer, right-click the predicate.
2. On the context menu, select **Delete Predicate**.
3. Click **Yes**. The predicate is deleted.

## Setting the Reset Destination Option

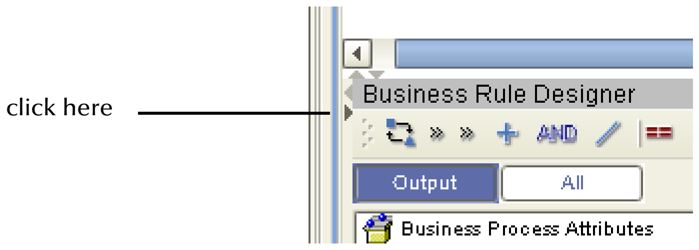
When you use predicates in a dynamic table, you might create a Page Flow in which the user can return to the Page Layout that contains the dynamic table. In this situation, you might want to “flush” the choices that the user made when the user previously accessed the Page Layout.

The following procedure describes how to set the appropriate option. Setting this option resets the values of the mapped nodes (that is, the destination) in the right side of the Business Rule Designer.

### To set the Reset destination option

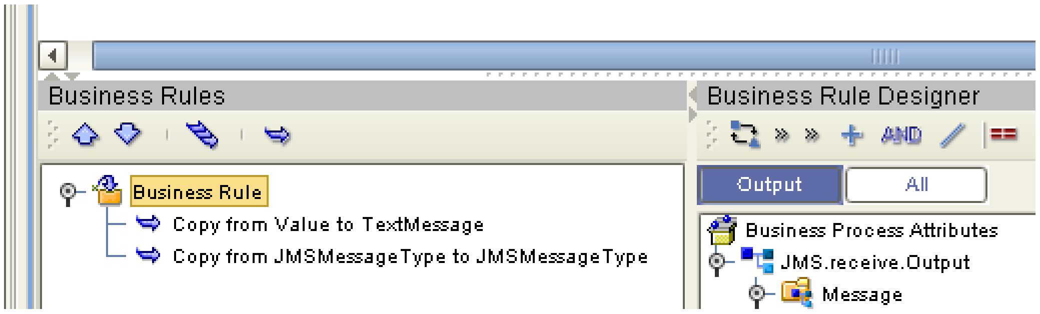
1. In the upper left corner of the Business Rule Designer, click the right-facing arrow. See Figure 79.

**Figure 79** Displaying the Business Rules Window



The **Business Rules** window appears. This window lists the names of each mapping in the Business Rule Designer. See Figure 80.

**Figure 80** Business Rules Window


Business Rules
Business Rule Designer
Business Rule
Copy from Value to TextMessage
Copy from JMSMessageType to JMSMessageType
Output
All
AND
Business Process Attributes
JMS.receive.Output
Message

2 Right-click the first business rule and turn on the **Reset destination** option. When this option is turned on, a check box appears next to the menu item.

3 Click the left-facing arrow. The **Business Rules** window closes.

## 6.5 Page Flow Properties

Each Page Flow has a set of properties. These properties enable the rapid creation and removal of Page Flow attributes. eVision uses this information to automatically create the appropriate Page Flow attributes and input/output structures, for use in the Business Rule Designer.

**To edit Page Flow properties**

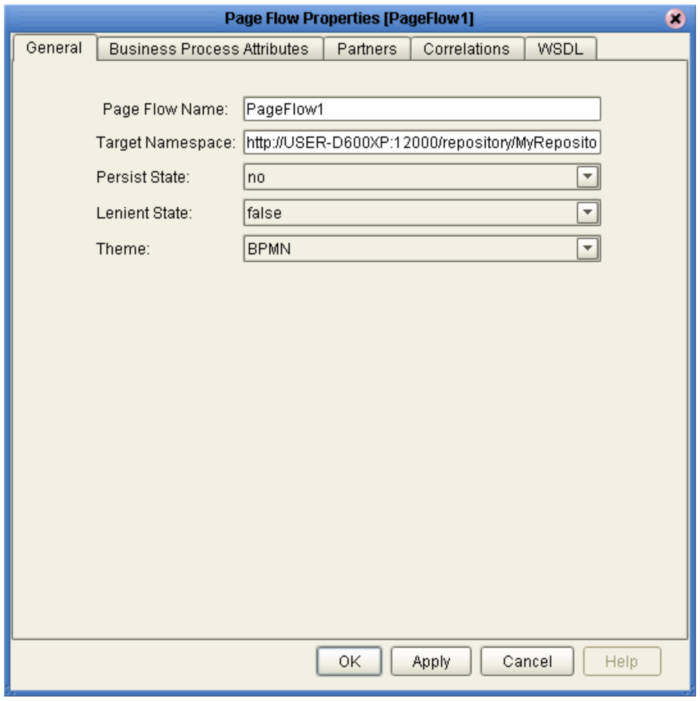
1. In the Project Explorer, right-click the Page Flow.
2. On the context menu, select **Properties**.

   The **Page Flow Properties** dialog box appears.
3. Select one or more tabs and edit properties. For more information, see the following subsections.

### 6.5.1. General Properties

The **General** tab allows you to edit general properties. See Figure 81.

**Figure 81** Page Flow Properties: General Tab



Page Flow Name: PageFlow1
Target Namespace: http://USER-D600XP:12000/repository/MyReposito
Persist State: no
Lenient State: false
Theme: BPMN

Table 11 describes the properties.

**Table 11** General Tab Properties

| Property | Description |
|---|---|
| Page Flow Name | The name of the Page Flow. |
| Target Namespace | The address of the Page Flow. |
| Persist State | Indicates whether the Page Flow will be persisted using a database. |
| Lenient State | This property applies only to Page Flows that are imported from ICAN Suite release 5.0.0, or third-party BPEL code that does not check for the presence of data before doing a copy.<br><br>If you incorrectly receive the error message **No pointer for xpath: *xpath***, then set the value for this property to true. |
| Theme | The look and feel of the Page Flow Designer. The default theme is **BPMN**. |

## 6.5.2. Page Flow Attributes

Page Flow Attributes are data values used by a Page Flow. They make it possible to share data between activities in a Page Flow as well as move data to and from the components that implement those activities. Complex structures such as Object Type Definitions (OTDs) and Collaborations are represented automatically in the Enterprise Explorer and are available for use in a Page Flow.

Some examples of Page Flow Attributes are:

- customer names
- addresses
- order quantities
- item descriptions

Page Flow Attributes are used to pass values between the Page Flow and external sources. You can assign Page Flow Attributes to specific activities. For example, the customer name is passed to an order process from the originating source. The customer name may be used by several of the activities in the Page Flow and is included in the Page Flow output.

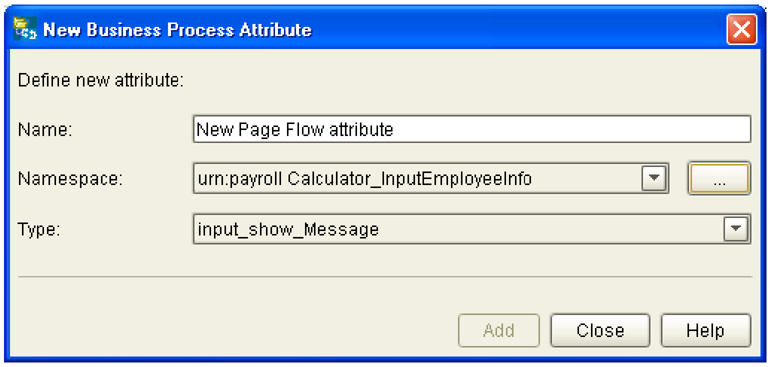
eVision can pass all or part of a complex structure, or it can even assemble a composite input to a component or Web service from multiple Page Flow attributes.

**To create a Page Flow attribute**

1. In the Project Explorer, right-click the Page Flow.
2. On the context menu, select **Properties**.
3. Select the **Page Flow Attributes** tab.
4. Click **Create**.

   The **New Page Flow Attribute** dialog box appears. See Figure 82.

**Figure 82** New Page Flow Attribute Dialog Box



New Business Process Attribute

Define new attribute:

| | |
|---|---|
| Name: | New Page Flow attribute |
| Namespace: | urn:payroll Calculator_InputEmployeeInfo |
| Type: | input_show_Message |

5 Do the following:

- Enter a **Name** for the attribute.
- Select or browse for an existing **Namespace**.
- Select an available **Type** for your attribute.

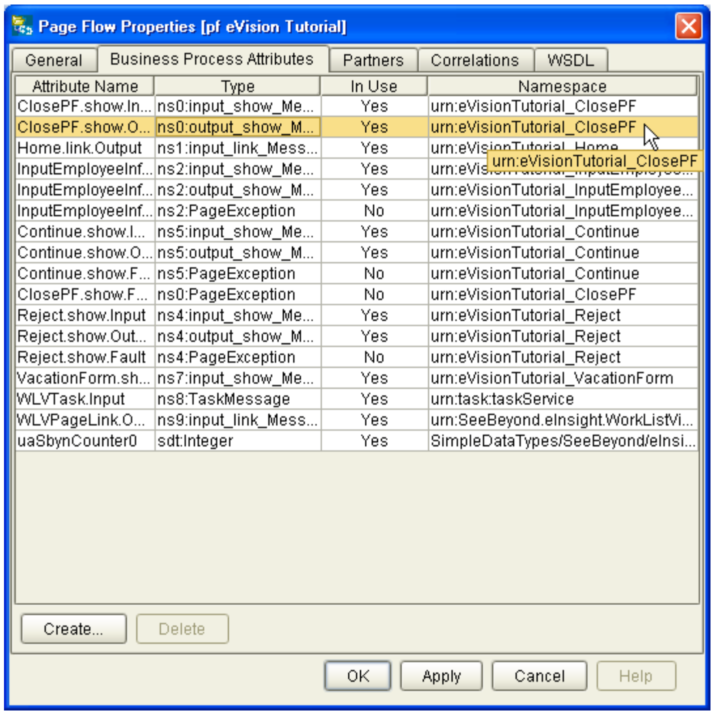
6 To save the attribute, click **Add**.

7 To return to the **Page Flow Properties** dialog box, click **Close**.

**To edit a Page Flow attribute**

1 On the **Page Flow Properties** dialog box, select the **Page Flow Attributes** tab. See Figure 83.

**Figure 83** Page Flow Properties: Page Flow Attributes tab



Page Flow Properties [pf eVision Tutorial]

| Attribute Name | Type | In Use | Namespace |
|---|---|---|---|
| ClosePF.show.In... | ns0:input_show_Me... | Yes | urn:eVisionTutorial_ClosePF |
| ClosePF.show.O... | ns0:output_show_M... | Yes | urn:eVisionTutorial_ClosePF |
| Home.link.Output | ns1:input_link_Mess... | Yes | urn:eVisionTutorial_Home |
| InputEmployeeInf... | ns2:input_show_Me... | Yes | urn:eVis urn:eVisionTutorial_ClosePF |
| InputEmployeeInf... | ns2:output_show_M... | Yes | urn:eVisionTutorial_InputEmployee... |
| InputEmployeeInf... | ns2:PageException | No | urn:eVisionTutorial_InputEmployee... |
| Continue.show.I... | ns5:input_show_Me... | Yes | urn:eVisionTutorial_Continue |
| Continue.show.O... | ns5:output_show_M... | Yes | urn:eVisionTutorial_Continue |
| Continue.show.F... | ns5:PageException | No | urn:eVisionTutorial_Continue |
| ClosePF.show.F... | ns0:PageException | No | urn:eVisionTutorial_ClosePF |
| Reject.show.Input | ns4:input_show_Me... | Yes | urn:eVisionTutorial_Reject |
| Reject.show.Out... | ns4:output_show_M... | Yes | urn:eVisionTutorial_Reject |
| Reject.show.Fault | ns4:PageException | No | urn:eVisionTutorial_Reject |
| VacationForm.sh... | ns7:input_show_Me... | Yes | urn:eVisionTutorial_VacationForm |
| WLVTask.Input | ns8:TaskMessage | Yes | urn:task:taskService |
| WLVPageLink.O... | ns9:input_link_Mess... | Yes | urn:SeeBeyond.eInsight.WorkListVi... |
| uaSbynCounter0 | sdt:Integer | Yes | SimpleDataTypes/SeeBeyond/eInsi... |

2 Select an existing attribute and do one of the following:

- To rename an attribute, double click the attribute text and type a new name. Some attributes cannot be renamed.
- To remove an attribute, select the attribute text and click **Delete**.

3 To save your changes and exit the **Page Flow Properties** dialog box, click **OK**.

## 6.5.3. Partners

A partner is an abstracted identification for an external system that will appear in the binding box in the Connectivity Map Editor. Multiple activities can use the same external system; therefore, multiple Activities may have the same Partner. By default, eVision assigns this identification to speed up and automate the model development.

## 6.5.4. Correlation Keys and Sets

To configure correlation, perform the following steps:

- **"Creating Correlation Keys" on page 107**
- **"Adding Correlation Sets" on page 109**
- **"Binding Correlation Sets to Page Flow Elements" on page 110**

### Creating Correlation Keys

A *correlation key* is a value that you can assign to a Page Flow, such as a Purchase Order number. The correlation key provides a way to associate and route information about specific Page Flow instances. For asynchronous message exchange between components, you must implement correlation of the instance identification. An example of when you use asynchronous message exchanges is when you create a Receive activity in the middle of a Page Flow.

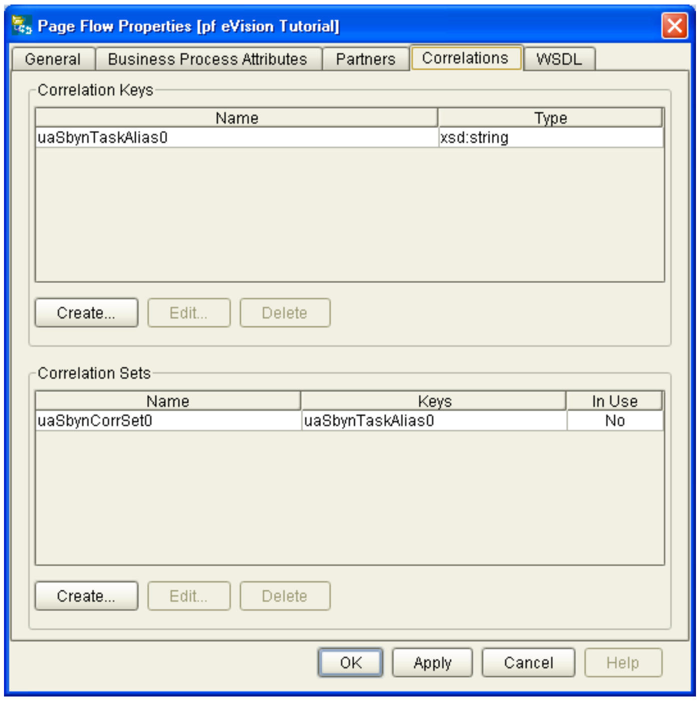
#### To create a correlation key

1 In the Project Explorer, right-click the Page Flow.

2 On the context menu, select **Properties**.

3 Select the **Correlations** tab. See Figure 84.

**Figure 84** Page Flow Properties: Correlations Tab


Page Flow Properties [pf eVision Tutorial]
General
Business Process Attributes
Partners
Correlations
WSDL
Correlation Keys

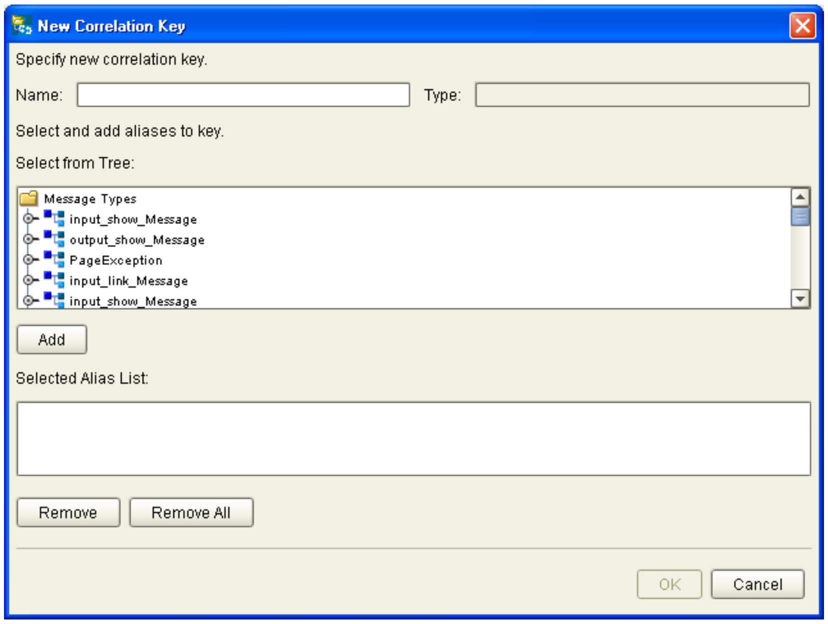
| Name | Type |
|---|---|
| uaSbynTaskAlias0 | xsd:string |

| Name | Keys | In Use |
|---|---|---|
| uaSbynCorrSet0 | uaSbynTaskAlias0 | No |

4 In the **Correlation Keys** section, click **Create**.

The **New Correlation Key** dialog box appears. See Figure 85.

**Figure 85** New Correlation Key Dialog Box



5 Enter a **Name** (alias) for the correlation key.

6 Select a **Message Type** from the list to alias. Select one or more correlation keys that comprise a unique identifier for a step in a Page Flow.

7 To save the new alias to the **Selected Alias List**, click **Add**.

8 To save your changes and exit the **New Correlation Key** dialog box, click **OK**.

## Adding Correlation Sets

*Correlation sets* are groups of properties shared by all messages in the group. A correlation set matches messages and conversations with a Page Flow instance. For example, you may want to assign a Purchase Order number and an invoice number to a transaction, so that all information about the purchase and payment are associated.

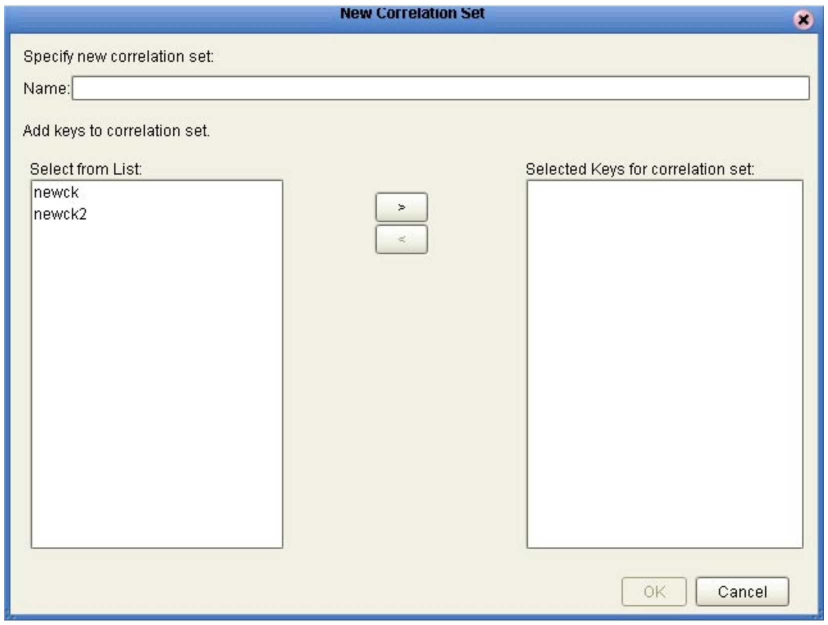
### To add a correlation set

1 Select the **Correlations** tab. See Figure 84.

2 In the **Correlation Sets** section, click **Create**.

The **New Correlation Set** dialog box appears. See Figure 86.

**Figure 86** New Correlation Set Dialog Box


New Correlation Set
Specify new correlation set:
Name:
Add keys to correlation set.
Select from List:
newck
newck2
>
<
Selected Keys for correlation set:
OK
Cancel

3 In the **Name** field, enter a name for the correlation set.

4 To add to the correlation set, select correlation keys from the list.

5 To move your selections to the correlation set, click the arrow button.

6 To save your changes and exit the **New Correlation Set** dialog box, click **OK**.

## Binding Correlation Sets to Page Flow Elements

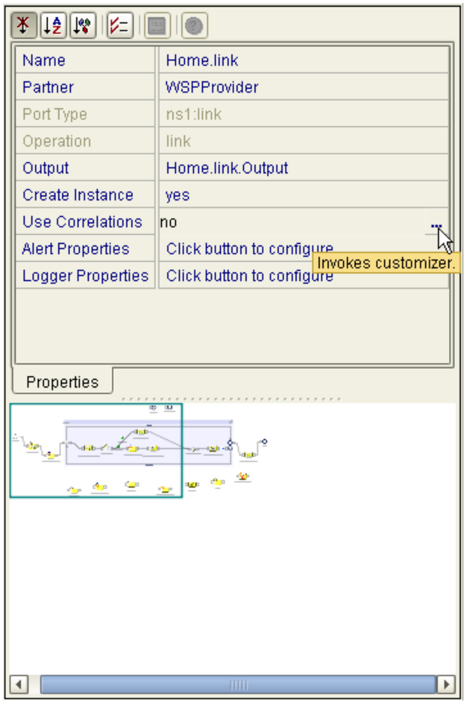
When you use one or more correlation sets in a Page Flow, the values must be initialized at some point. If the user chooses to initialize the set within an Activity, they will also identify which Page Flow Attribute will be used (or both).

### To bind correlation sets to Page Flow elements

1 Select a Page Flow element.

2 On the Page Flow Designer toolbar, click the **Show Property Sheet** icon. The properties window appears. See Figure 87.

**Figure 87** Page Flow Element Properties Window



| | |
|---|---|
| Name | Home.link |
| Partner | WSPProvider |
| Port Type | ns1:link |
| Operation | link |
| Output | Home.link.Output |
| Create Instance | yes |
| Use Correlations | no |
| Alert Properties | Click button to configure |
| Logger Properties | Click button to configure |

3 Locate the **Use Correlations** property and click the **no** field.

4 In the **no** field, click the **Command (...)** button. The **Use Correlations** dialog box appears.

5 Click **Add**. The **Assign Correlation Set** dialog box appears.

6 In the left pane, select the correlation set that you want to add to the Page Flow element.

7 Click the arrow button to move it to **Selected Correlation Set(s)** area.

8 Click **OK**.

9 In the **Use Correlations** dialog box, click **OK**.

### 6.5.5. WSDL Files

Web Services Description Language (WSDL) is an XML-based language used to describe business services. WSDL provides a way for individuals and other businesses to electronically access those services. In the Page Flow Designer, WSDL files are used to invoke and operate Web services on the Internet and to access and invoke remote applications and databases.

The WSDL tab is available from the **Page Flow Properties** dialog box. You can upload a WSDL file with predefined Page Flow Attributes for use in your Page Flow.

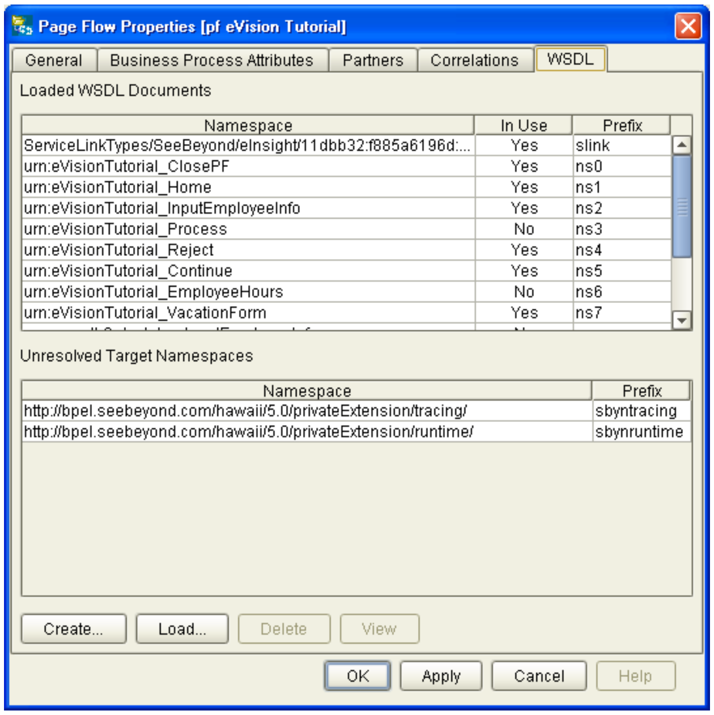
#### To upload a WSDL file

1 In the Project Explorer, right-click the Page Flow.

2 On the context menu, select **Properties**.

3 Select the **WSDL** tab. See Figure 88.

**Figure 88** Page Flow Properties: WSDL Tab



Loaded WSDL Documents

| Namespace | In Use | Prefix |
|---|---|---|
| ServiceLinkTypes/SeeBeyond/eInsight/11dbb32:f885a6196d:... | Yes | slink |
| urn:eVisionTutorial_ClosePF | Yes | ns0 |
| urn:eVisionTutorial_Home | Yes | ns1 |
| urn:eVisionTutorial_InputEmployeeInfo | Yes | ns2 |
| urn:eVisionTutorial_Process | No | ns3 |
| urn:eVisionTutorial_Reject | Yes | ns4 |
| urn:eVisionTutorial_Continue | Yes | ns5 |
| urn:eVisionTutorial_EmployeeHours | No | ns6 |
| urn:eVisionTutorial_VacationForm | Yes | ns7 |

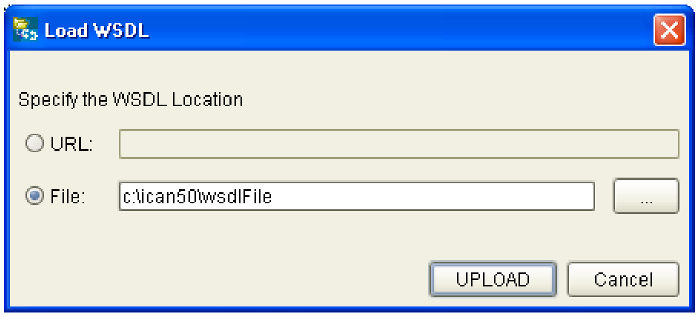
Unresolved Target Namespaces

| Namespace | Prefix |
|---|---|
| http://bpel.seebeyond.com/hawaii/5.0/privateExtension/tracing/ | sbyntracing |
| http://bpel.seebeyond.com/hawaii/5.0/privateExtension/runtime/ | sbynruntime |

4 To upload a WSDL file, click **Load**.

The **Load WSDL** dialog box appears. See Figure 89.

**Figure 89** Load WSDL Dialog Box


Load WSDL

Specify the WSDL Location

URL:

File: c:\ican50\wsdlFile

5 To specify the location of your WSDL, select **URL** or **File**.

6 In the text field, type the path to the WSDL file.

7 Click **UPLOAD**.

## 6.6 Page Flows in Connectivity Maps

When you create the Connectivity Map for a Project that contains one or more Page Flows, you must add each Page Flow in the Project to a Service. You then link each Service to a Web Connector.

### 6.6.1. Adding Each Page Flow to a Service

For each Page Flow in the Project, create a Service and then drag the Page Flow from the Project Explorer into the Service. The red gears in the Service change to a Page Flow symbol. See Figure 90.

**Figure 90** Change in Appearance of Service

Before Page Flow is added
Service1
After Page Flow is added
Service1

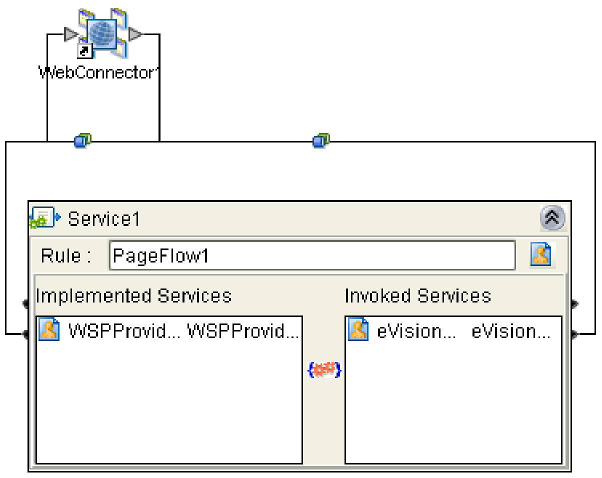
As a shortcut, you can drag the Page Flow from the Project Explorer without creating a service.

### 6.6.2. Linking Each Service to a Web Connector

The Web Connector is a logical representation of the Web container in which an eVision Studio Web application runs.

Drag a Web Connector from the Connectivity Map Editor toolbar onto the canvas. Double-click each Service that contains a Page Flow. A binding box appears. Link the **WSPProvider** Implemented Service to the Web Connector. Link the **eVision_user** Invoked Service to the Web Connector. See Figure 91.

**Figure 91** Service Binding Box

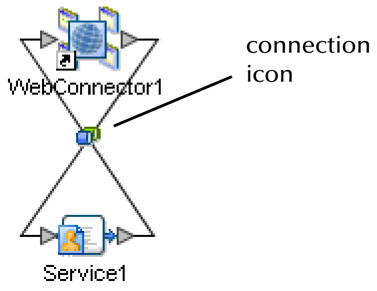


Implemented Services represent the Web services that are implemented and thus served by the Page Flow. The **WSPProvider** service is the entry point (the home page) of the Web application.

Invoked Services represent the Web services that are called by the Page Flow.

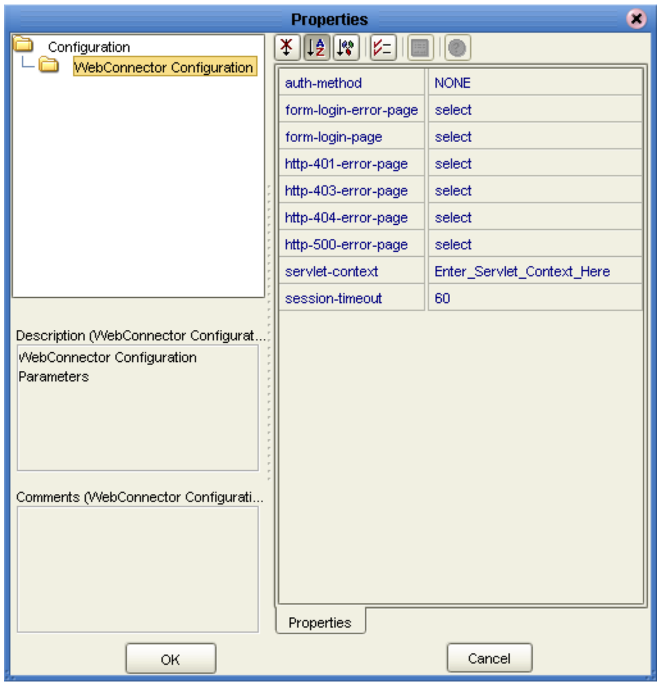
When you close the binding box, the connectors will appear crossed. This is normal. In addition, a connection icon appears on the link. See Figure 92.

**Figure 92** Linked Service and Web Connector



The Web application consists of a presentation component and a back-end component. To configure properties for the presentation component of the application, open the binding box and double-click the connection icon between the **WSPProvider** Implemented Service and the Web Connector. The **Properties** dialog box appears.

**Figure 93** Web Connector Configuration Properties


Properties
Configuration
WebConnector Configuration
Description (WebConnector Configurat...
Comments (WebConnector Configurati...
Properties
OK
Cancel

Description (WebConnector Configurat...: WebConnector Configuration Parameters

| | |
|---|---|
| auth-method | NONE |
| form-login-error-page | select |
| form-login-page | select |
| http-401-error-page | select |
| http-403-error-page | select |
| http-404-error-page | select |
| http-500-error-page | select |
| servlet-context | Enter_Servlet_Context_Here |
| session-timeout | 60 |

Chapter 9, **"Authentication and Error Handling"** describes the following properties:

- auth-method
- form-login-error-page
- form-login-page
- http-401-error-page
- http-403-error-page
- http-404-error-page
- http-500-error-page

**"Application URL" on page 115** describes the **servlet-context** property.

The **session-timeout** property is expressed in a whole number of minutes.

## 6.7 Deploying Page Flows

The general steps for deploying a Project are:

1. Create an Environment.
2. Create and activate a Deployment Profile.
3. Start the Logical Host.

The *eGate Integrator User's Guide* describes the first two steps. The *eGate Integrator System Administration Guide* describes the third step.

This section contains information that is specific to deploying Projects that contain Page Flows.

### 6.7.1. Application URL

To access an eVision Studio Web application, users enter the *application URL* in the address field of their browser.

The application URL has the form:

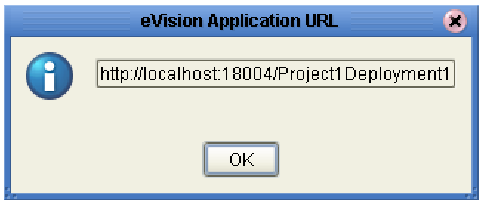
http://***hostname***:***portnumber***/***servletcontext***

For example:

**http://avalon:18004/Project1Deployment1**

When you activate the Deployment Profile, the **eVision Application URL** dialog box displays the application URL. See Figure 94.

**Figure 94** eVision Application URL Dialog Box



http://localhost:18004/Project1Deployment1

***Note:*** *The* ***eVision Application URL*** *dialog box allows you to copy and paste the URL.*

#### Hostname

The hostname portion of the application URL represents the computer where the Integration Server is run.

In the **eVision Application URL** dialog box, the hostname is derived from the Integration Server's **Web Server Host Name** property. The default value for this property is **localhost**.

If the Integration Server is run on a computer other than the local host, then the hostname in the **eVision Application URL** dialog box will not be correct. However, users will still be able to access the Web application using the hostname of the computer where the Integration Server is run.

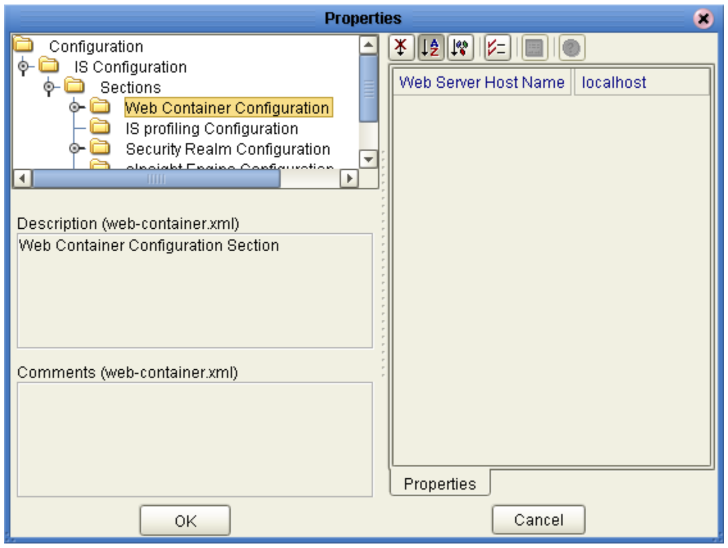
If you want to ensure that the **eVision Application URL** dialog box displays the correct hostname, perform the following steps before activation:

1 In the Environment Explorer of Enterprise Designer, right-click the Integration Server.

2 On the context menu, select **Properties**.

3 Expand the tree and select **Web Container Configuration**. See Figure 95.

**Figure 95** Integration Server Properties - Web Container Configuration



Properties
Configuration
IS Configuration
Sections
Web Container Configuration
IS profiling Configuration
Security Realm Configuration
Properties
OK
Cancel

Description (web-container.xml)
Web Container Configuration Section

Comments (web-container.xml)

| | |
|---|---|
| Web Server Host Name | localhost |

4 Set the value of the **Web Server Host Name** property to the hostname of the computer where the Integration Server is run.

5 Click **OK**.

## Servlet Context

The default value of the servlet context is the Project name concatenated with the Deployment Profile name. If you want to change the servlet context to a more user-friendly value, perform the following steps before activation:

1 In the Connectivity Map Editor, open the binding box and double-click the connection icon between the **WSPProvider** Implemented Service and the Web Connector.

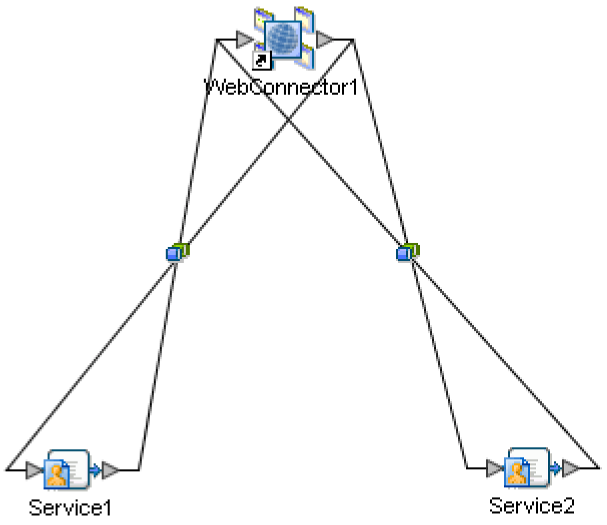
2 Change the value of the **servlet-context** property.

3 Click **OK**.

## Multiple Web Applications

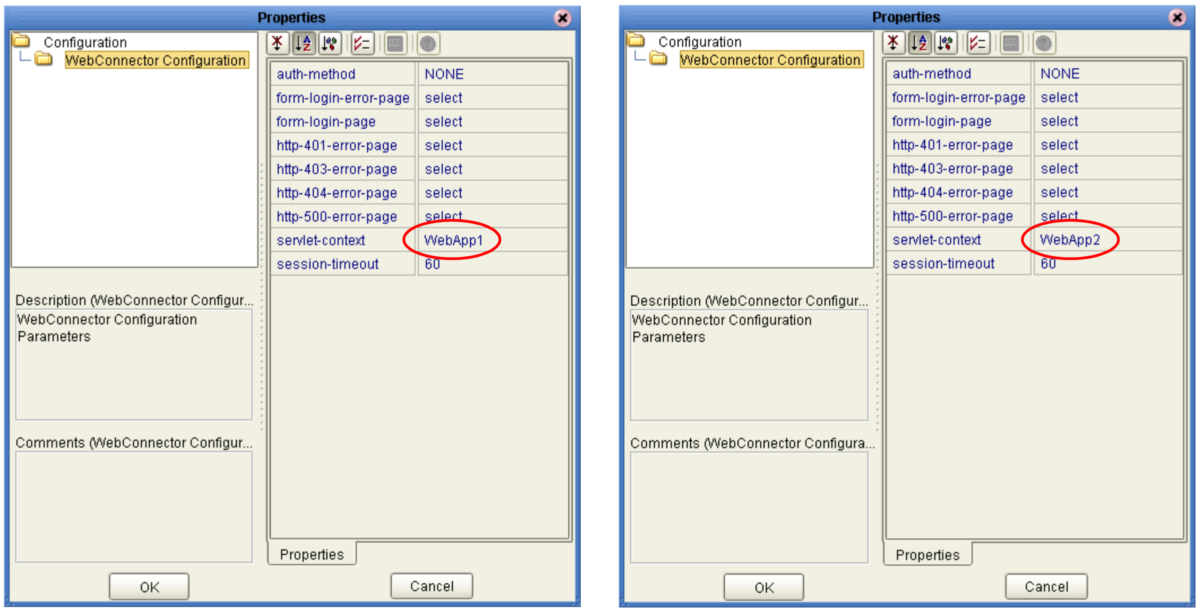
eVision support multiple Web applications in the same deployment. See Figure 96.

**Figure 96** Connectivity Map with Multiple Web Applications



You must configure each application to have a different URL. To do this, open the connection icons and ensure that the **servlet-context** properties have different values. See Figure 97.

**Figure 97** servlet-context Properties for Multiple Web Applications



Description (WebConnector Configur...
WebConnector Configuration Parameters

Comments (WebConnector Configur...

| | |
|---|---|
| auth-method | NONE |
| form-login-error-page | select |
| form-login-page | select |
| http-401-error-page | select |
| http-403-error-page | select |
| http-404-error-page | select |
| http-500-error-page | select |
| servlet-context | WebApp1 |
| session-timeout | 60 |

Description (WebConnector Configur...
WebConnector Configuration Parameters

Comments (WebConnector Configura...

| | |
|---|---|
| auth-method | NONE |
| form-login-error-page | select |
| form-login-page | select |
| http-401-error-page | select |
| http-403-error-page | select |
| http-404-error-page | select |
| http-500-error-page | select |
| servlet-context | WebApp2 |
| session-timeout | 60 |

If you do not perform this step, then the generated Web application will not deploy, because multiple **.war** files will conflict with the same servlet context.

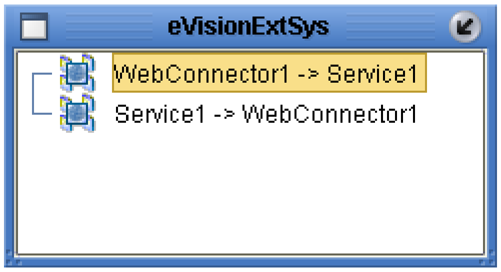
## 6.7.2. eVision External System

When you create the Environment, you must include an eVision External System. The eVision External System is a logical representation of the Web container in which an eVision Studio Web application runs. It is where the presentation component of the application will execute.

The eVision External System contains a **user-role** property, which is used in authentication. For more information, see Chapter 9, **"Authentication and Error Handling"**.

When you create the Deployment Profile, some of the components in the left panel of the Deployment Profile Editor will contain a Web Connector. Drag these components into the eVision External System. See Figure 98.

**Figure 98** Web Connector Components in eVision External System



Chapter 7

# Page Flow Designer Tutorial

This chapter guides you through the process of creating and deploying a Web application.

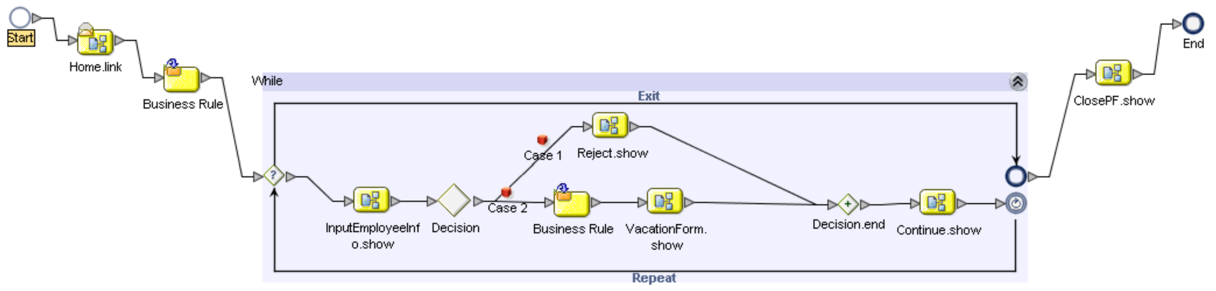
**This Chapter Includes:**

## 7.1 Overview

In the following tutorial, you import, assemble, and run the sample Web application that is packaged with eVision Studio. Going through the Page Flow creation process allows you to work with several of the most-used Page Flow Designer tools while assembling the necessary components to complete a Web application that you can deploy, then run on the Logical Host, and access with your browser.

Figure 99 shows the Page Flow Designer view of the sample application, which allows employees to request vacation time.

**Figure 99** Page Flow Designer View of Sample Application



## 7.2 Downloading the Sample Project

Download the sample project only if you have not already done so. If you created the sample Web page as described in **Chapter 5**, **"Page Flow Designer Tutorial"**, then you already downloaded the files. Skip to **"Importing the Sample Project into the Repository" on page 121**.

The sample project is provided in **.zip** file format. It is included in the eVision Studio documentation **.sar** file. The first step in this tutorial is to download the sample project from the Repository.

The Repository must be running before you can use Enterprise Manager or Enterprise Designer. The first step of any procedure is to make sure that the Repository is running. The *SeeBeyond ICAN Suite Installation Guide* describes how to start the Repository.

### To download the sample project

1. Start Internet Explorer.
2. In the **Address** field, enter **http://*hostname*:*portnumber***

   where

   ***hostname*** is the TCP/IP host name of the server where the Repository is installed.

   ***portnumber*** is the base port number of the Repository.

   The **SeeBeyond Customer Login** window of Enterprise Manager appears.
3. Enter your username and password.
4. Click **Login**.

   The Enterprise Manager home page appears.
5. Click the **DOCUMENTATION** tab.
6. In the left frame of the Documentation page, click **eVision Studio**.

**Note:** *If you do not see an eVision Studio link in the left frame, then the eVision Studio Documentation* ***.sar*** *file must be uploaded. See* **Chapter 3**, **"Installing eVision Studio" on page 25**.

7. In the right frame of the Documentation page, click **Download Sample**.
8. Using an archive utility (such as WinZip), extract the sample files to a folder on your computer. Remember where you saved the files. If you use the default folder names, the sample files will be stored in a directory called **eVision_Sample**.

   The **eVisionSampleComponents.zip** file contains the pre-built Web pages that you will use to create the sample application.

   The **eVisionTutorial_sample.zip** file is for reference and analysis *after* you successfully build the sample application from the pre-built components.

## 7.3 Importing the Sample Project into the Repository

You import the **eVisionSampleComponents.zip** Project using Enterprise Designer. Make sure that the Repository is running before you start Enterprise Designer.

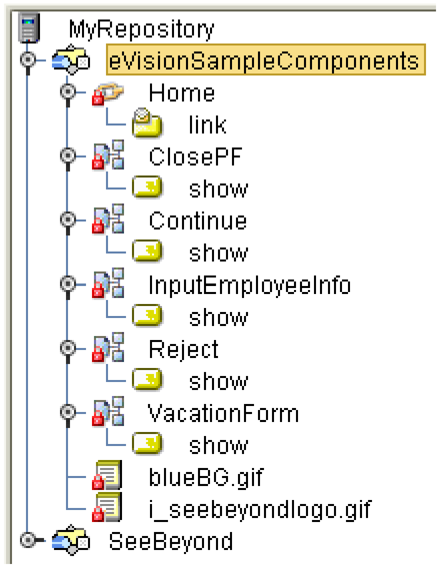
**Note:** *Do not import the* ***eVisionTutorial_sample.zip*** *Project until you finish this tutorial.*

**To import the sample project into the Repository**

1. Start Enterprise Designer.
2. In the Project Explorer, right click the Repository.
3. On the context menu, select **Import**. The **Import** dialog box appears.
4. Assuming that you have saved any changes to other Projects, click **Yes**. Import Manager appears.
5. Click **Browse** and navigate to the folder that contains the extracted sample files.
6. Select **eVisionSampleComponents.zip** and click **Open**.
7. Click **Import**.

   When the import process is finished, a confirmation message appears.
8. Click **OK** and then click **Close**. The Repository refreshes.
9. In the Project Explorer, expand the Project to reveal the components. The Project contains a Page Link, five Page Layouts, and two image files. See Figure 100.

**Figure 100** Sample Project Components in Project Explorer



## 7.4 Checking Out the Project Components

The icon for each component includes a red lock, which means that the component is checked into the Version Control system. Before you can use the components, you must check them out one at a time.

**To check out the Project components**

1. In the Project Explorer, under **eVisionSampleComponents**, right-click the **Home** Page Link.
2. On the context menu, click **Check Out**. The **Version Control - Check Out** dialog box appears.
3. Click **OK**. The red lock disappears, and a writing pad icon appears.
4. Repeat these steps for each component.

## 7.5 Creating the Page Flow

This section describes how to assemble the eVision components to create a Page Flow.

Because the Page Link and the Page Layouts are provided in the sample project, you do not need to create them.

You will use elements in the Page Flow Designer's toolbar to connect the components in a Page Flow. These elements are described in detail in Chapter 6, **"Using the Page Flow Designer"**. See Figure 101.

**Figure 101** Page Flow Designer Toolbar

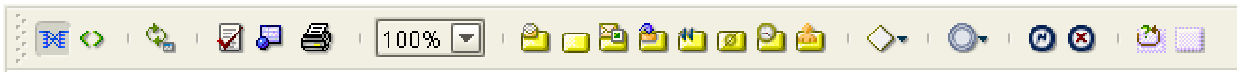


Table 12 describes the Page Layouts in the sample project. You can open each Page Layout and examine it. However, do not modify any of the Page Layouts.

**Table 12** Page Layouts in Sample Project

| Page Layout | Description |
|---|---|
| ClosePF | Indicates that the vacation request has been submitted. The user can exit the application. |
| Continue | Enables the user to continue or exit. |
| InputEmployeeInfo | Enables the user to enter his or her name and employee number, and then click a Request Vacation button. |
| Reject | Indicates that authorization of the employee information failed. The user can click a button to try again. |
| VacationForm | Enables the user to enter the number of vacation days and the manager's e-mail address, and then click a Submit button. |

## 7.5.1 Starting a New Page Flow

**To start a new Page Flow**

1. In the Project Explorer, right-click **eVisionSampleComponents**.
2. On the context menu, click **New**, and then select **Page Flow**.

   A new Page Flow icon appears in the Project Explorer tree under **eVisionSampleComponents**, and a blank Page Flow appears on the Page Flow Designer canvas (the right pane of Enterprise Designer). The default name is **PageFlow1**. A new Page Flow always contains both **Start** and **End** elements.

## 7.5.2 Adding the Page Flow Elements

In this procedure, you add the Page Flow elements to the Page Flow.

**To add the initial Page Flow elements**

1. In the Project Explorer, expand the **Home** Page Link, if necessary.
2. From the Project Explorer, drag the **Home.link** operation onto the canvas.

   Place the **Home.link** icon to the right of and below the **Start** element.
3. From the Page Flow Designer toolbar, drag a **Business Rule** element onto the canvas.

   Place the **Business Rule** element to the right of and below the **Home.link** icon.

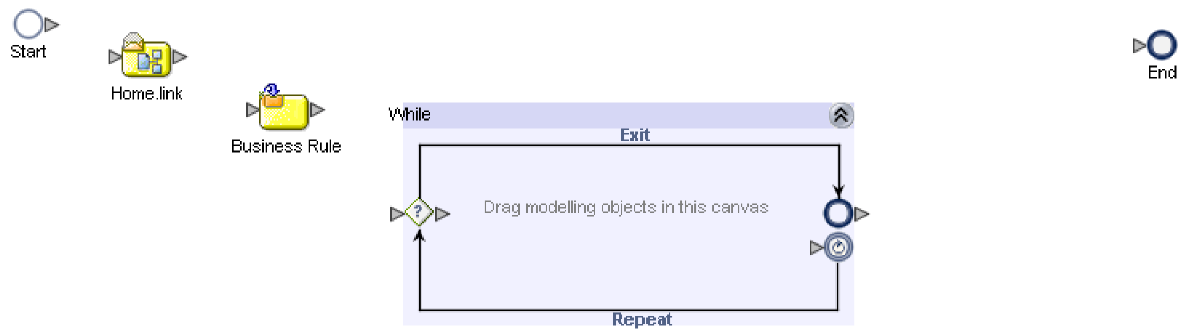
4 From the Page Flow Designer toolbar, drag a **While** element onto the canvas.

Place the **While** icon between the **Business Rule** element and the **End** element.

5 Double-click the **While** icon to expand it.

6 Check your progress. The Page Flow should look like the example in Figure 102.

**Figure 102** Initial Page Flow Elements



### To add the Page Flow elements inside the While loop

1 From the Project Explorer, drag the **InputEmployeeInfo.show** operation onto the canvas.

Without releasing the mouse button, place the **InputEmployeeInfo.show** icon to the far left inside the **While** Loop boundaries.

***Caution:*** *Do not place the icon outside of the **While** Loop and then drag the icon inside the **While** Loop.*

2 On the Page Flow Designer toolbar, click the **Branching Activities** icon. On the drop-down menu, click **Decision**, and drag the **Decision** element onto the canvas.

Without releasing the mouse button, place the **Decision** element to the right of and aligned with the **InputEmployeeInfo.show** icon inside the **While** loop.

Two icons appear: **Decision** and **Decision.end**.

3 Move the **Decision.end** icon to the right so that you have enough room to place two icons between **Decision** and **Decision.end**. Notice how the **While** loop becomes wider.

4 From the Page Flow Designer toolbar, drag a **Business Rule** element onto the canvas.

Without releasing the mouse button, place the **Business Rule** element to the right of and aligned with the **Decision** element inside the **While** loop.

5 From the Project Explorer, drag the **VacationForm.show** operation onto the canvas.

Without releasing the mouse button, place the **VacationForm.show** icon to the right of and aligned with the **Business Rule** element inside the **While** loop.

6 From the Project Explorer, drag the **Continue.show** operation onto the canvas.

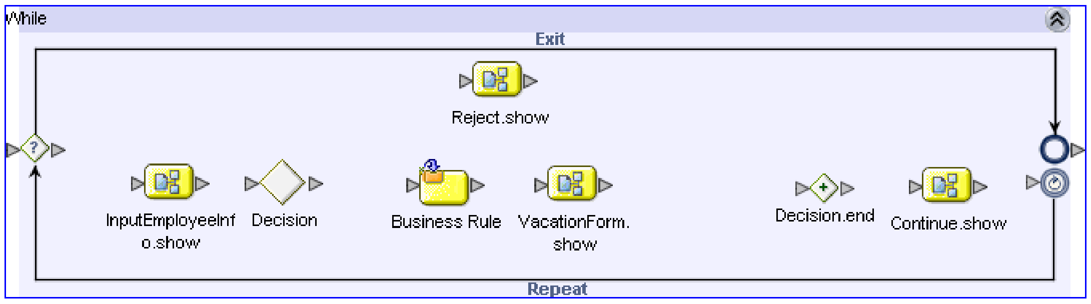
Without releasing the mouse button, place the **Continue.show** icon to the right of and aligned with the **Decision.end** element inside the **While** loop.

7 From the Project Explorer, drag the **Reject.show** operation onto the canvas.

Without releasing the mouse button, place the **Reject.show** icon above the other elements and center it in the **While** loop. Notice how the **While** loop becomes taller.

8 Check your progress. The **While** loop should look like the example in Figure 103.

**Figure 103** Page Flow Elements Inside the While Loop



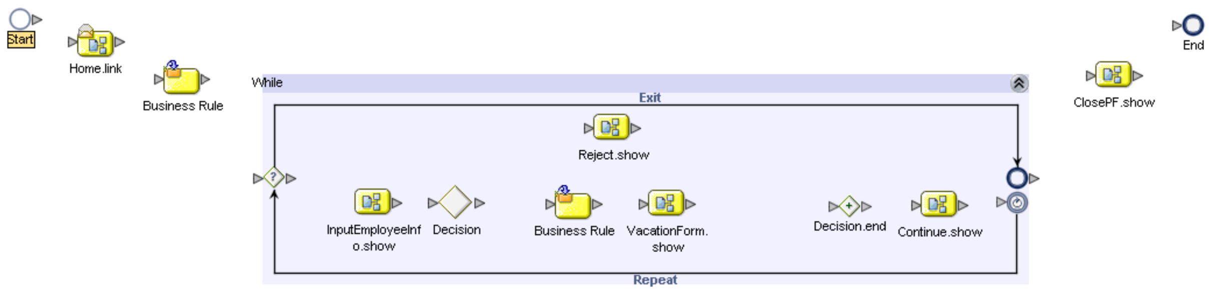
### To add the final Page Flow element

1 From the Project Explorer, drag the **ClosePF.show** operation onto the canvas.

Place the **ClosePF.show** icon between the **While** loop and the **End** element.

2 Check your progress. The Page Flow should look like the example in Figure 104.

**Figure 104** Unconnected Page Flow Components



## 7.5.3 Connecting the Page Flow Elements

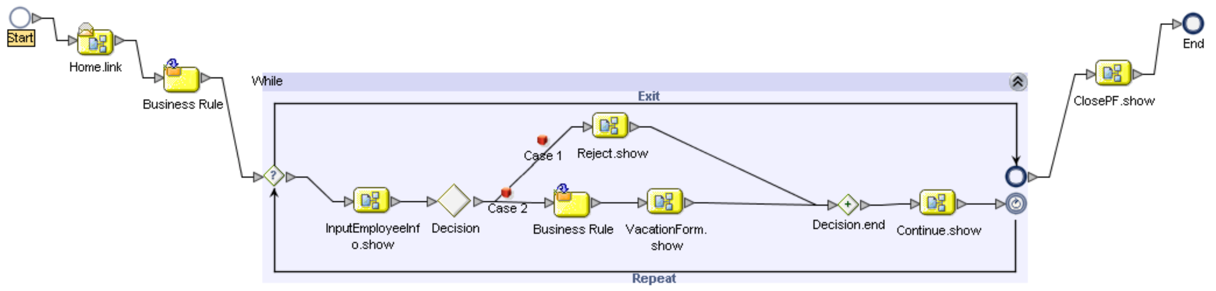
In this procedure, you link the Page Flow elements that you added to the Page Flow.

**To connect the Page Flow elements**

1. Hover your mouse pointer over the arrow on the right edge of the **Start** icon until the pointer changes to the hand symbol.
2. Click and drag a connection line onto the **Home.link** element and release the mouse button.
3. Proceeding from left to right, connect all of the Page Flow components. See Figure 105.

   Note that the **Decision** element has two branching links. To ensure that Case 1 is on the top and Case 2 is on the bottom, create the link between the **Decision** element and the **Reject.show** icon and then create the link between the **Decision** element and the **Business Rule** element.

**Figure 105** Connected Page Flow Components



4. On the **File** menu, click **Save All**.

## 7.5.4 Configuring the First Business Rule Element

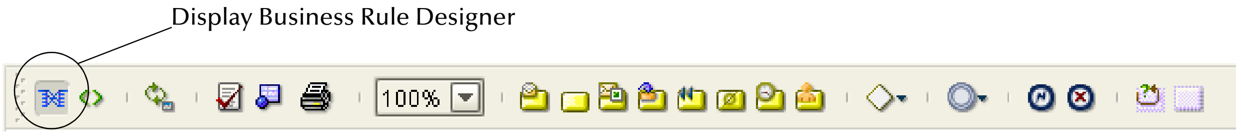
In the following procedures, you use the Business Rule Designer to configure relationships between Output and Input Attributes. The Business Rule Designer consists of three panes:

- **Output** (on the left)
- **Mapper** (in the center)
- **Input** (on the right)

**To configure the first Business Rule element**

1. Click the **Display Business Rule Designer** icon on the Page Flow Designer toolbar. See Figure 106.

**Figure 106** Displaying the Business Rule Designer



The Business Rule Designer appears in the lower pane of the Page Flow Designer.

2 On the Page Flow Designer canvas, select the **Business Rule** element that resides outside and to the left of the **While** loop.

***Caution:*** *Do not select the **Business Rule** element that resides inside the **While** loop.*

The Business Rule Designer is populated with Output and Input Attributes.

3 In the Business Rule Designer **Input** pane (the right pane), expand **Continue.show.Output** to expose the **radiogroup0** node.

The **radiogroup0** node represents the **Radio Group** component in the **Continue** Page Layout. **radiogroup0** is the value of the component's **lname** property.

4 Expand the **radiogroup0** node to expose the **value** element.

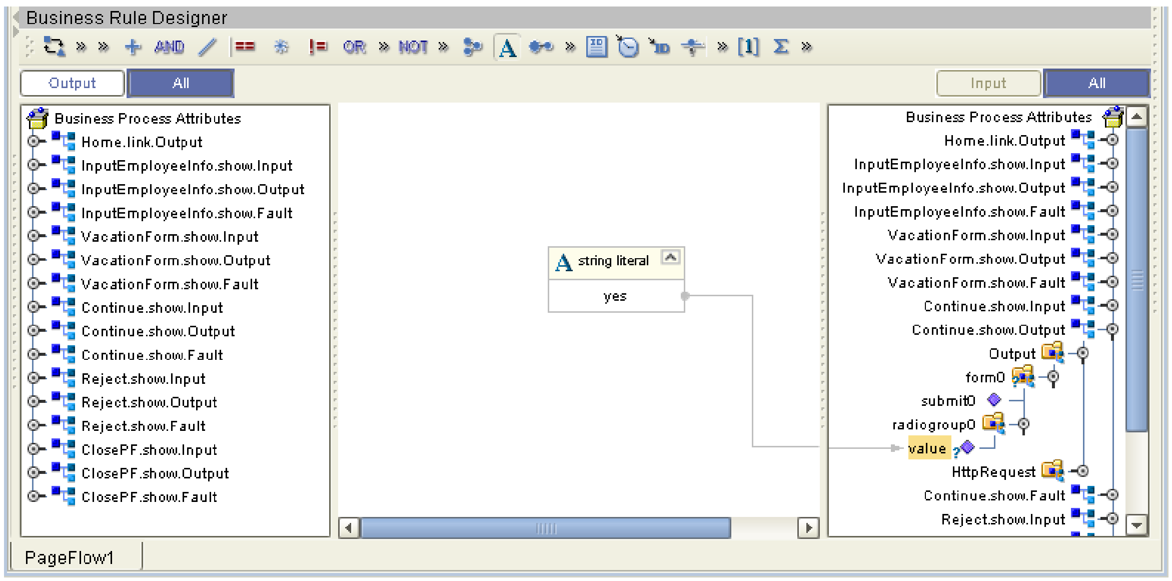
5 From the Business Rule Designer Method Palette, drag the **string literal** icon onto the Mapper. The **Input** dialog box appears.

6 In the **Enter a Literal Value** box, type **yes**. Make sure that the text is lower-case.

7 Click **OK**.

8 Select the **string literal** container within the **yes** box, drag it into the **Input** pane, and drop it onto the **value** element. See Figure 107.

**Figure 107** Mapping a String Literal



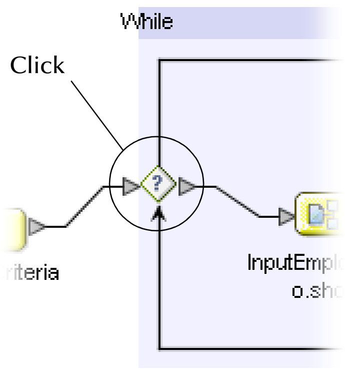
### 7.5.5 Configuring the While Loop

In this procedure, you specify the boolean expression that controls whether the **While** loop is executed.

**To configure the While loop**

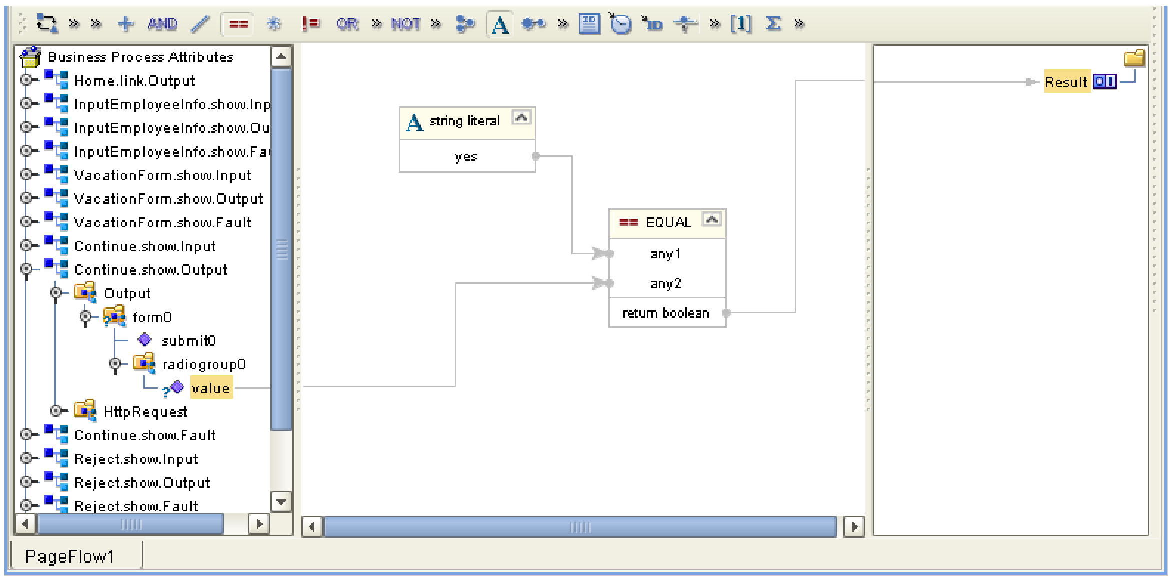
1. At the left boundary of the **While** loop, click the diamond with a question mark inside it. See Figure 108.

**Figure 108** While Loop Icon



2. In the Business Rule Designer **Output** pane (the left pane), expand the **Continue.show.Output** node to expose the **radiogroup0** node.
3. Expand the **radiogroup0** node to expose the **value** element.
4. From the Business Rule Designer Method Palette, drag the **string literal** icon onto the Mapper. The **Input** dialog box appears.
5. In the **Enter a Literal Value** box, type **yes**. Make sure that the text is lower-case.
6. Click **OK**.
7. From the Business Rule Designer Method Palette, drag the **EQUAL** icon (a "double-equal" sign) onto the Mapper.
8. Click the **string literal** container within the **yes** box and drag the cursor onto the **any1** connector node on the **EQUAL** container.
9. From the **Output** pane, click the **radiogroup0 value** node and drag it onto the **any2** connector node on the **EQUAL** container.
10. Click the **EQUAL** container on the **return boolean** operator window, drag the **return boolean** element into the **Input** pane, and drop it onto the **Result** icon. See Figure 109.

**Figure 109** Specifying a Boolean Expression for the While Loop



## 7.5.6 Configuring the Decision Logic

A **Decision Gate** element must have its properties defined such that, based on incoming events, there can be two or more path possibilities within the Page Flow. In this procedure, you define properties for a **Decision Gate** element that contains two cases.

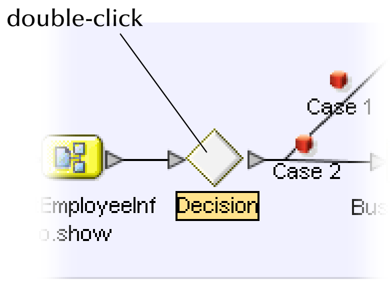
### Opening the Decision Gate Properties Window

You configure the decision logic from the **Decision Gate Properties** window.

**To open the Decision Gate Properties window**

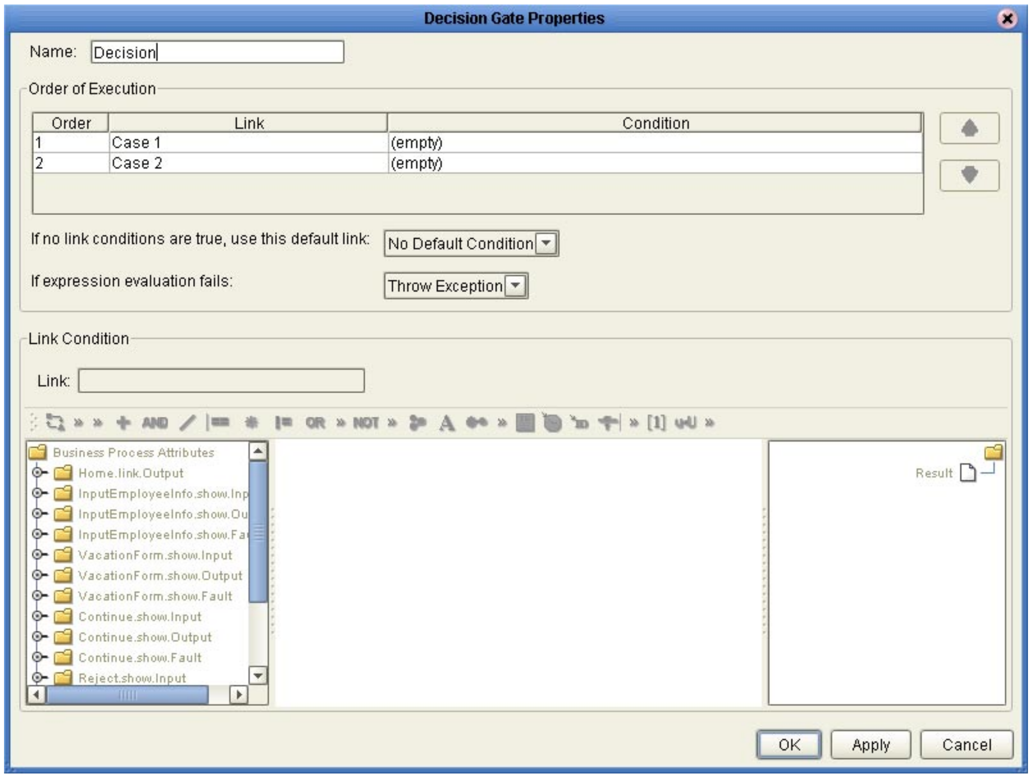
1 On the Page Flow Designer canvas, double-click the **Decision** element inside the **While** loop. See Figure 110.

**Figure 110** Decision Element in While Loop



The **Decision Gate Properties** window appears. See Figure 111.

**Figure 111** Decision Gate Properties Window



Decision Gate Properties

Name: Decision

Order of Execution

| Order | Link | Condition |
|---|---|---|
| 1 | Case 1 | (empty) |
| 2 | Case 2 | (empty) |

If no link conditions are true, use this default link: No Default Condition

If expression evaluation fails: Throw Exception

Link Condition

Link:

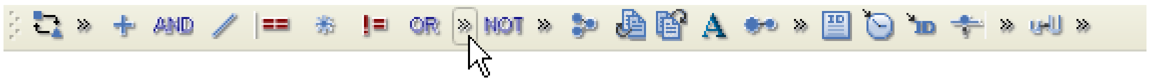
Business Process Attributes
Home.link.Output
InputEmployeeInfo.show.Inp
InputEmployeeInfo.show.Ou
InputEmployeeInfo.show.Fa
VacationForm.show.Input
VacationForm.show.Output
VacationForm.show.Fault
Continue.show.Input
Continue.show.Output
Continue.show.Fault
Reject.show.Input

2 Under the **Order of Execution** heading, click **Case 1** to activate the Mapper.

3 When you define the properties for Case 1, you use the **greater than** function in the Method Palette. If the **greater than** function does not currently appear on the toolbar, perform the following steps:

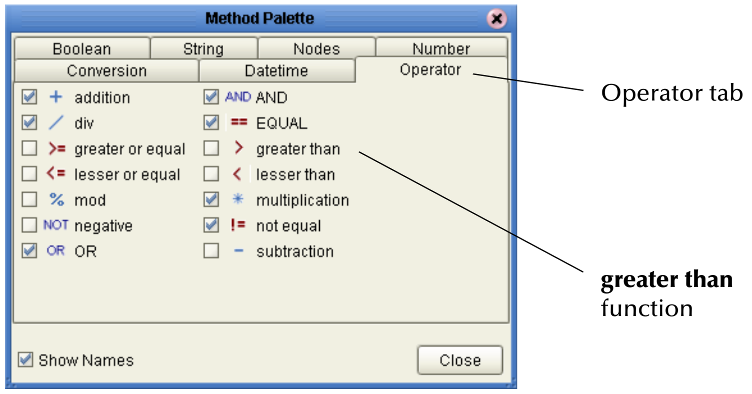
A To launch the Method Palette dialog box, click the chevrons to the right of the **OR** operator. See Figure 112.

**Figure 112** Launching the Method Palette Dialog Box



B Ensure that the **Operator** tab is selected. See Figure 113.

**Figure 113** Method Palette Dialog Box with Operator Tab Selected



C Click the **greater than** check box to add it to the toolbar on the **Decision Gate Properties** window.

D Click **Close**.

## Defining the Properties for Case 1

Case 1 is triggered when the user enters an invalid employee number in the **InputEmployeeInfo** Page Layout. Any number greater than 5000 is invalid.

**To define the properties for Case 1**

1 From the **Decision Gate Properties** toolbar, drag the **string literal** icon onto the Mapper. The **Input** dialog box appears.

2 In the **Enter a Literal Value** box, type **5000**. This number represents the highest valid employee number.

3 Click **OK**.

4 From the **Decision Gate Properties** toolbar, drag the **greater than** icon onto the Mapper.

5 In the **Output** pane (the left pane) of the **Decision Gate Properties** window, expand the **InputEmployeeInfo.show.Output** node until the **EmpNumber** node appears.

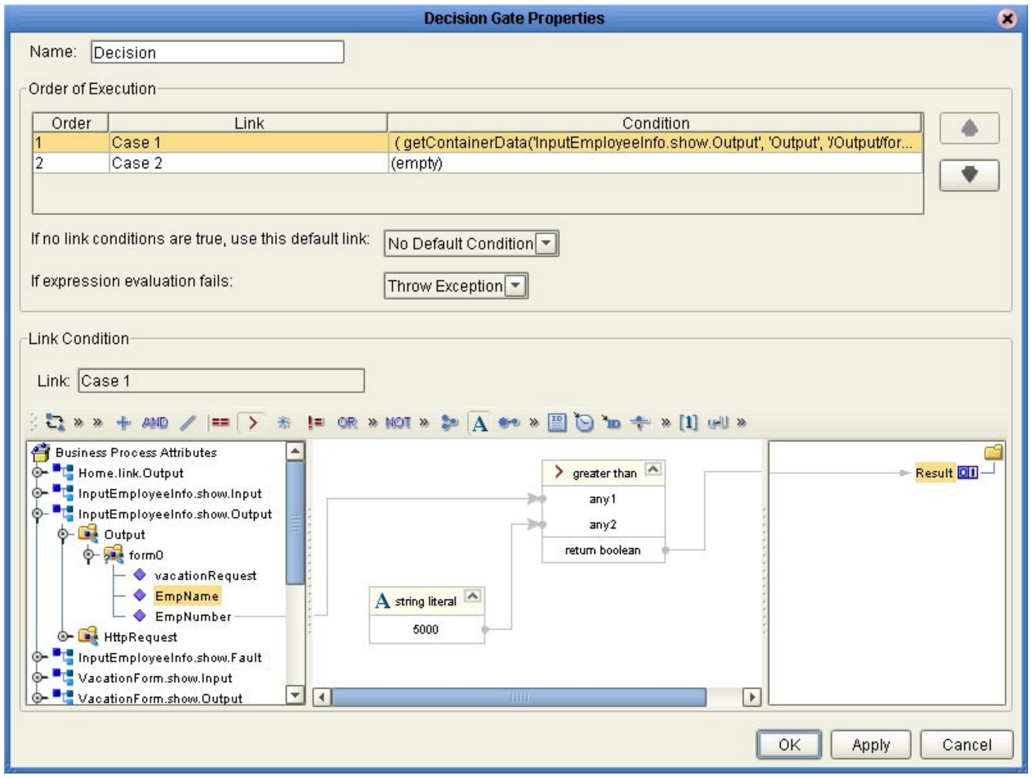
The **EmpNumber** node represents the text box to the right of the **Enter Employee Number** label in the **InputEmployeeInfo** Page Layout. **EmpNumber** is the value of the text box component's **lname** property.

6 Drag the **EmpNumber** node onto the Mapper and, on the **greater than** container, drop it onto the **any1** node.

7 Click the **string literal** container near the center of the value box (the value is 5000) and, on the **greater than** container, drop it onto the **any2** node.

8 Click the **greater than** container near the center of the **return boolean** box, drag it into the **Input** pane (the right pane), and drop it onto the **Result** icon. See Figure 114.

**Figure 114** Defining Decision Gate Properties: Case 1



| Order | Link | Condition |
|---|---|---|
| 1 | Case 1 | ( getContainerData('InputEmployeeInfo.show.Output', 'Output', '/Output/for... |
| 2 | Case 2 | (empty) |

## Defining the Properties for Case 2

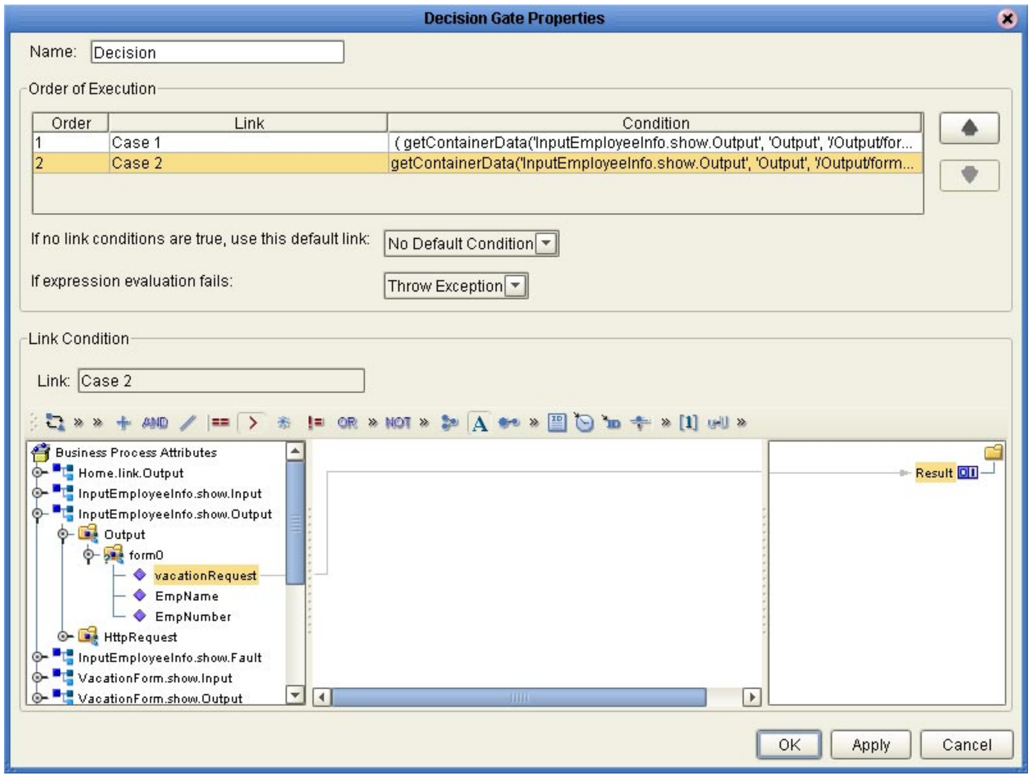
Case 2 is triggered when the user enters a valid employee number in the **InputEmployeeInfo** Page Layout. Any number less than or equal to 5000 is valid.

### To define the properties for Case 2

1. Under the **Order of Execution** heading, click **Case 2**.
2. In the **Output** pane (the left pane), expand the **InputEmployee.show.Output** node until the **vacationRequest** node is exposed.

   The **vacationRequest** node represents the **Request Vacation** button in the **InputEmployeeInfo** Page Layout. **vacationRequest** is the value of the component's **lname** property.
3. Click the **vacationRequest** node, drag it into the **Input** pane (the right pane), and drop it onto the **Result** icon. See Figure 115.

**Figure 115** Defining Decision Gate Properties: Case 2



Decision Gate Properties

Name: Decision

Order of Execution

| Order | Link | Condition |
|---|---|---|
| 1 | Case 1 | ( getContainerData('InputEmployeeInfo.show.Output', 'Output', '/Output/for... |
| 2 | Case 2 | getContainerData('InputEmployeeInfo.show.Output', 'Output', '/Output/form... |

If no link conditions are true, use this default link: No Default Condition

If expression evaluation fails: Throw Exception

Link Condition

Link: Case 2

4 Click **Apply**.

5 Click **OK**. The **Decision Gate Properties** window closes.

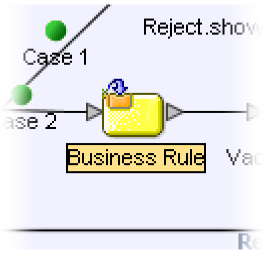
## 7.5.7 Mapping the Employee Name to the Vacation Form

In this procedure, you map the employee name that the user entered in the **InputEmployeeInfo** Page Layout to the **VacationForm** Page Layout.

**To map the employee name to the vacation form**

1 On the Page Flow Designer canvas, click the **Business Rule** element inside the **While** loop. See Figure 116.

**Figure 116** Business Rule Element in the While Loop



2 In the Business Rule Designer **Output** pane (the left pane), expand the **InputEmployeeInfo.show.Output** node to expose the **EmpName** node.

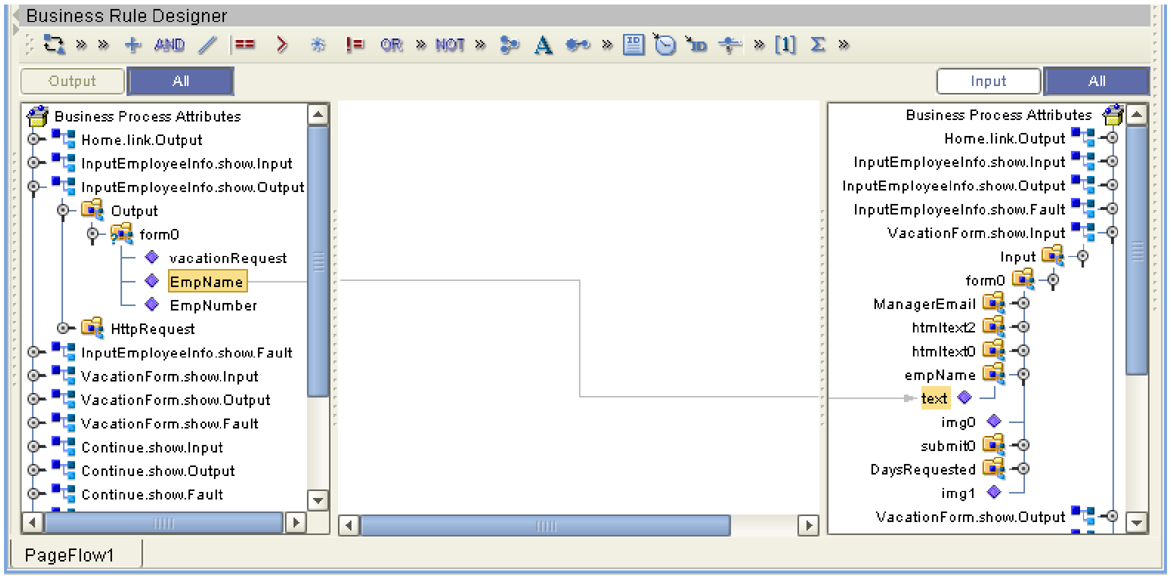
The **EmpName** node represents the text box to the right of the **Enter Name** label in the **InputEmployeeInfo** Page Layout. **EmpName** is the value of the text box component's **lname** property.

3 In the Business Rule Designer **Input** pane (the right pane), expand the **VacationForm.show.Input** node to expose the **empName text** node.

The **empName** node represents the upper **HTML Text** component in the **VacationForm** Page Layout. **empName** is the value of the component's **lname** property.

4 Click the **EmpName** node, drag it onto the **Input** pane, and drop it onto the **text** node. See Figure 117.

**Figure 117** Mapping the Employee Name to the Vacation Form



## 7.5.8 Restoring the Input Values

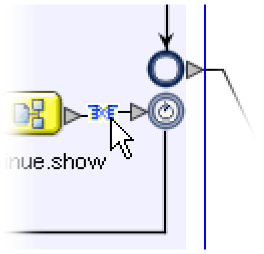
This is the last mapping procedure. In the Web application, when the user returns to the **InputEmployeeInfo** Page Layout, the input values are restored.

### To restore the input values

1 On the Page Flow Designer canvas, right-click the connector between the **Continue.show** page and the **While** loop.

2 On the context menu, click **Add Business Rule**. See Figure 118.

**Figure 118** Adding a Business Rule



3 In the **Output** pane, click **All**.

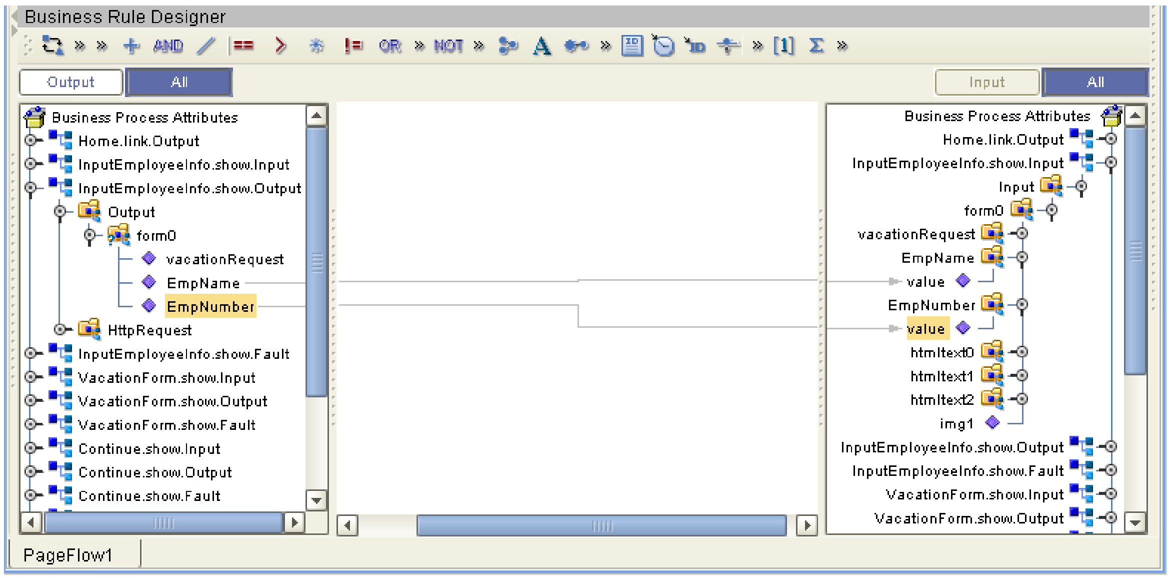
4 In the **Output** pane, expand the **InputEmployeeInfo.show.Output** node to expose the **EmpName** and **EmpNumber** nodes.

5 In the **Input** pane, expand the **InputEmployeeInfo.show.Input** node to expose the **value** nodes under the **EmpName** and **EmpNumber** nodes.

6 In the **Output** pane, click the **EmpName** node, drag it into the **Input** pane, and drop it onto the **value** node under the **EmpName** node.

7 In the **Output** pane, click the **EmpNumber** node, drag it into the **Input** pane, and drop it onto the **value** node under the **EmpNumber** node. See Figure 119.

**Figure 119** Mapping the Employee Name and Employee Number



8 On the **File** menu, click **Save All**.

The Page Flow is complete.

## 7.6 Creating the Connectivity Map

The Connectivity Map enables the connections between the system components and the back-end systems.

The Connectivity Map contains an eGate **Service** and a **Web Connector**. These components enable the project to run on the Logical Host. The Web Connector is a logical representation of the Web container in which the Web application runs.

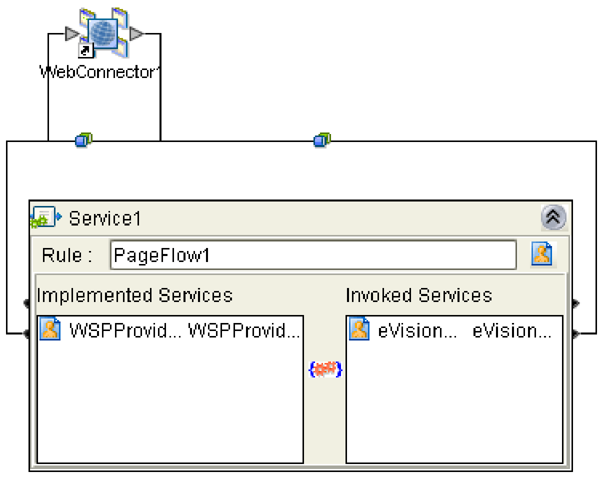
### To create the Connectivity Map

1. In the Project Explorer, right-click **eVisionSampleComponents**.
2. On the context menu, click **New**, and then select **Connectivity Map**.

   A new Connectivity Map node appears in your Project. The default name is **CMap1**. The Connectivity Map Editor appears.
3. On the Connectivity Map Editor toolbar, select the **Service** icon and drag it onto the canvas.
4. In the Project Explorer, select the **PageFlow1** icon and drag it onto the **Service** icon on the canvas.

   The **Service** icon changes to reflect containment of the Page Flow.
5. On the Connectivity Map Editor toolbar, select the **Web Connector** icon and drag it onto the canvas above the **Service** icon.
6. Double-click the **Service** icon. A binding box appears.
7. Connect the **WSPProvider** Implemented Service to the **Web Connector** icon. Connect the **eVision_user** Invoked Service to the **Web Connector** icon. See Figure 120.

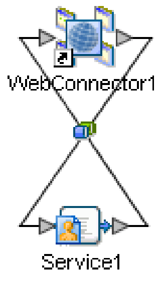
**Figure 120** Connecting the Service to the Web Connector



The connection icon between the **WSPProvider** Implemented Service and the Web Connector enables you to configure properties for the presentation component of the application. In this tutorial, you do not need to configure any of the properties.

8 Close the binding box. The connectors will appear crossed. This is normal.

**Figure 121** Crossed Appearance of Connectors



## 7.7 Creating the Environment

In this procedure, you create the run-time environment for the Web application. The environment consists of:

- A Logical Host (the run-time platform)
- A SeeBeyond Integration Server (provides run-time services for the application)
- An eVision External System (represents where the presentation component of the application will execute)

**To create the Environment**

1 On the **View** menu, click **Environment Explorer**.

2 Right-click the Repository icon and on the context menu, select **New Environment**.

The default name of the Environment is **Environment*n***, where ***n*** is the next sequential default Environment number. You can accept the default name.

3 Right-click the Environment and on the context menu, select **New Logical Host**.

The default name of the Logical Host is **LogicalHost1**. You can accept the default name.

4 Right-click the Logical Host and on the context menu, select **New SeeBeyond Integration Server**.

The default name of the Integration Server is **IntegrationSvr1**. You can accept the default name.

5 Right-click the Environment and on the context menu, select **New eVision External System**.

6 When prompted, enter **eVisionExtSys** and click **OK**.

7 On the **File** menu, click **Save All**.

## 7.8 Creating and Activating the Deployment Profile

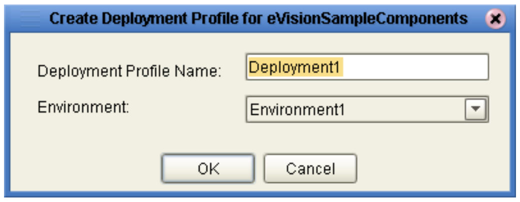
Before you can deploy the Web application to the Environment, you must create and activate a Deployment Profile.

### To create the Deployment Profile

1. On the **View** menu, click **Project Explorer**.
2. In the Project Explorer, right-click **eVisionSampleComponents**.
3. On the context menu, click **New**, and then select **Deployment Profile**.

   The **Create Deployment Profile** dialog box appears. See Figure 122.

   **Figure 122** Create Deployment Profile Dialog Box

   Create Deployment Profile for eVisionSampleComponents

   Deployment Profile Name: Deployment1

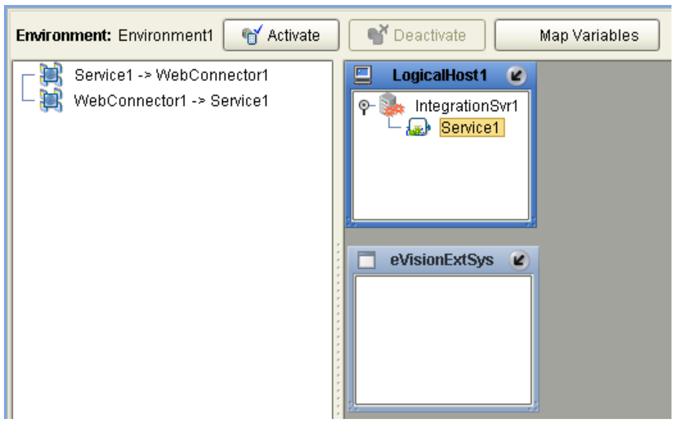
   Environment: Environment1

   The default name of the Deployment Profile is **Deployment1**. You can accept the default name.

   Ensure that the **Environment** drop-down list is set to the Environment that you just created.

4. Click **OK**. The Deployment Editor appears.
5. Click the **Service1** icon in the left panel, drag it into the **LogicalHost1** window, and drop it onto the **IntegrationSvr1** icon. See Figure 123.

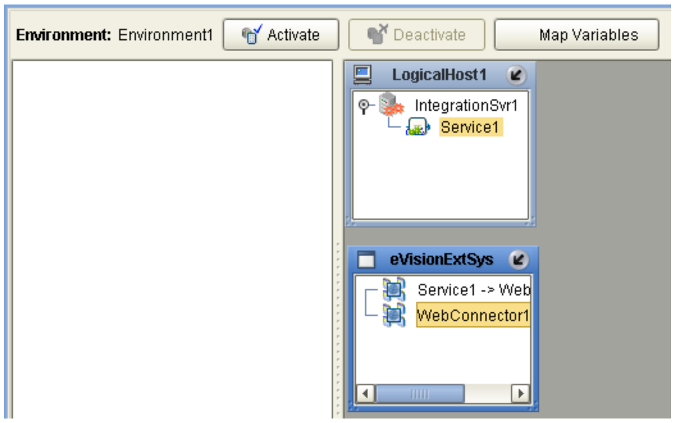
   **Figure 123** Service1 Icon in LogicalHost1 Window

   Environment: Environment1

6. One at a time, drag both Web Connector icons from the left panel into the **eVisionExtSys** window. See Figure 124.

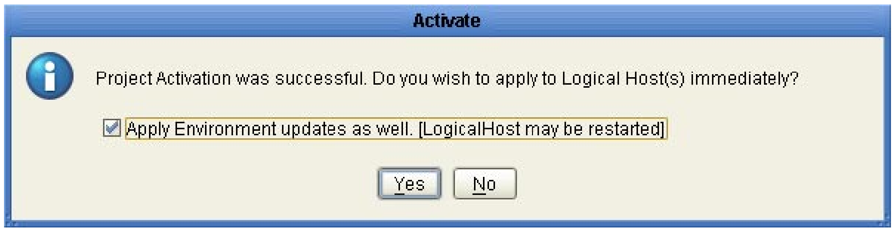
**Figure 124** Web Connector Icons in eVisionExtSys Window



### To activate the Deployment Profile

1 Click **Activate**. After a while, the **Activate** dialog box appears. See Figure 125.

**Figure 125** Activate Dialog Box



Activate

Project Activation was successful. Do you wish to apply to Logical Host(s) immediately?

☑ Apply Environment updates as well. [LogicalHost may be restarted]

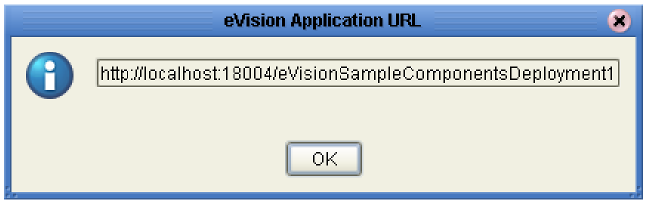
2 Click **No**.

You click **No** because the Logical Host is not yet running. If you click **Yes**, the operation will fail. You will start the Logical Host in another operation.

Once a project is deployed and you modify it and then re-deploy it, you can click **Yes** (if the Logical Host is already running).

3 The **eVision Application URL** dialog box displays the URL for the Web application that you just created. You will need to enter the URL in the address field of your browser to access the Web application. Note that the **Application URL** dialog box allows you to copy and paste the URL. See Figure 126.

**Figure 126** eVision Application URL Dialog Box



eVision Application URL

http://localhost:18004/eVisionSampleComponentsDeployment1

4 Click **OK**.

## 7.9 Running and Testing the Application

Now that you have activated the Deployment Profile, you can start the Logical Host and access the Web application in your browser.

### 7.9.1 Starting the Logical Host

If you do not have a Logical Host installed, follow the instructions in the *SeeBeyond ICAN Suite Installation Guide* to install one.

The Logical Host bootstrap process executes the **eVisionSampleComponents** Project and starts the process of polling for input data. The bootstrap process will pick up the Deployment Profile the first time it runs; after that, you select **Reactivate** and click **Yes** to apply the most recent changes to the Logical Host.

To start the Logical Host, you run a bootstrap script. The syntax is:

```
bootstrap argument1 ... argumentN
```

Some of the arguments are required, and other arguments are optional. The following table describes the required arguments.

**Table 13** Logical Host Bootstrap Required Arguments

| Argument | Description |
|---|---|
| -e *environment name* | The name of the Environment to which this Logical Host belongs. |
| -l *logicalhost name* | The name of the Logical Host. |
| -r *repository URL* | The root URL of the Repository containing the Logical Host data. |
| -i *username* | The username for accessing the Repository. |
| -p *password* | The password for accessing the Repository. |

The following procedure assumes that the Logical Host is running on a Windows system.

**To start the Logical Host**

1. Open a command prompt.
2. Navigate to the ***ICAN-root*\logicalhost\bootstrap\bin** directory.
3. Run the bootstrap script with the required arguments. For example:

```
bootstrap -e Environment1 -l LogicalHost1
-r http://frodo.acme.com:12000/MyRepository
-i Administrator -p STC
```

4. Wait until a message appears indicating that the Logical Host is ready.

### 7.9.2 Accessing the Web Application

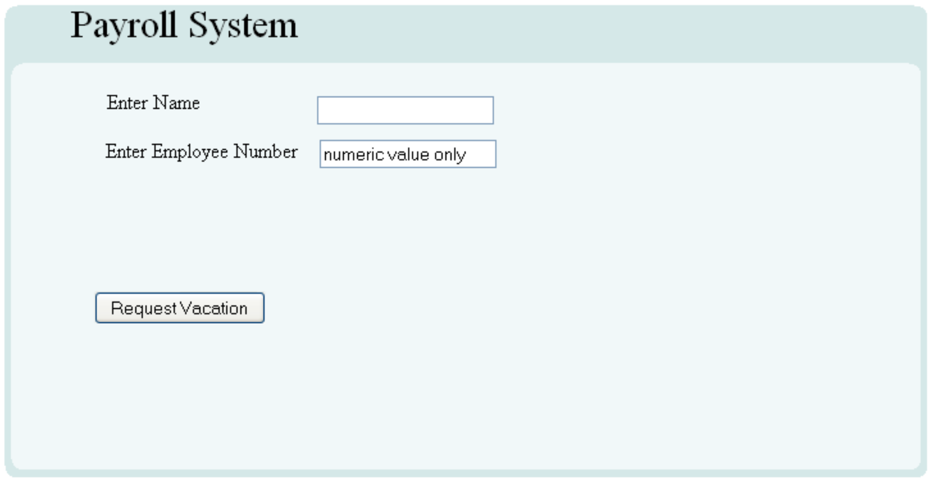
With the application running on the Logical Host, you can now access it in your browser.

**To access the Web application**

1. Ensure that the Repository is running.
2. Start your Web browser.
3. Enter the application URL.

The first page should look like the example in Figure 127.

**Figure 127** Sample Web Application Startup Page



Enter an employee name and an employee number less than or equal to 5000, and then click **RequestVacation**. The vacation form prompts you to enter the desired number of days and your manager’s e-mail address.

On the first page, you can try entering an employee number greater than 5000. A page with an error message will prompt you to try again.

## 7.10 Importing the Working Sample Project

When you are done with the tutorial, you can examine the sample reference Project in the **eVisionTutorial_sample.zip** file.

The Project is called **eVisionTutorial**. The Project contains two Page Flows. Each Page Flow has a corresponding Connectivity Map. Therefore, the Project enables you to deploy two Web applications. See Table 14.

**Table 14** Page Flows and Connectivity Maps in eVisionTutorial Project

| Page Flow | Connectivity Map | Description |
|---|---|---|
| pf eVision Tutorial | cm eVision Tutorial | This application is a completed version of the application that you created in this chapter. |
| pfTutorial2 | pfTutorialcmap | This application is a more complex version of the application that you created in this chapter. In addition to requesting vacation time, employees can submit data for their timecard. |

To deploy either application, you must import the Project, create an Environment, create and activate a Deployment Profile, and start a Logical Host.

***Note:*** *When you import the Project, ensure that the Project name is **eVisionTutorial**. If you use another name, some of the links may break when you open it in the Page Flow Designer.*

When you create a Deployment Profile, the components for both Page Flows appear in the left panel. You can do one of the following:

- Deploy the components for the **pf eVision Tutorial** Page Flow
- Deploy the components for the **pfTutorial2** Page Flow
- Deploy the components for both Page Flows

If you deploy the components for both Page Flows, two versions of the **eVision Application URL** dialog box will appear.

The application URL for the **pf eVision Tutorial** Page Flow will end with **testPF**.

You can access the applications in your browser, and work through the finished applications from a user’s perspective. You can also modify the Project to experiment with various features.

Chapter 8

# Creating Charts

This chapter describes how to use the Chart GUI component. This component is available from the **Form Objects** palette in the Page Layout Designer.

**This Chapter Includes:**

## 8.1 Overview

eVision provides a variety of predefined chart types, including area charts, bar charts, line charts, and pie charts.

The process of creating a chart is divided into two phases:

1. Adding the chart to a Page Layout in the Page Layout Designer
2. Mapping data into the chart in the Page Flow Designer

## 8.2 Adding a Chart to a Page Layout

The Page Layout Designer allows you to add one or more charts to a Page Layout.

**To add a chart to a Page Layout**

1. From the **Form Objects** palette, drag the **Chart** component onto the canvas.

   An area chart with boilerplate information appears.
2. If you want to change the chart type, do the following in the **Properties** sheet:
   - A In the left column, click the **type** property.
   - B In the right column, select the chart type from the drop-down menu. **"Chart Types" on page 144** describes the available chart types.
3. Specify the behavior of each data set. See **"Data Sets" on page 151**.
4. Set additional properties as needed. See **"Additional Properties" on page 152**.

5 To preview the chart, click the **Preview** icon on the Page Layout Designer toolbar. This feature has the following limitations:

- Because you have not mapped data into the chart, you will see only boilerplate information. The actual data will appear at runtime.
- You cannot see the values of the **xAxisLabel** and **yAxisLabel** properties until runtime.
- You cannot see the color specified by the **bgColor** property until runtime.

## 8.2.1 Chart Types

This section describes the available chart types.

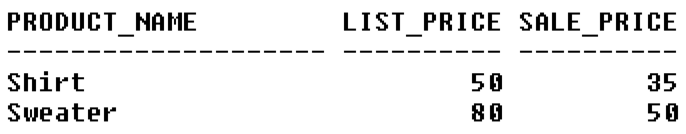
### Area

An area chart fills in the portion of the chart between the category axis and the lines that connect the data points. eVision provides the following area chart types:

- **areaChart**
- **stackedAreaChart**
- **xyAreaChart**

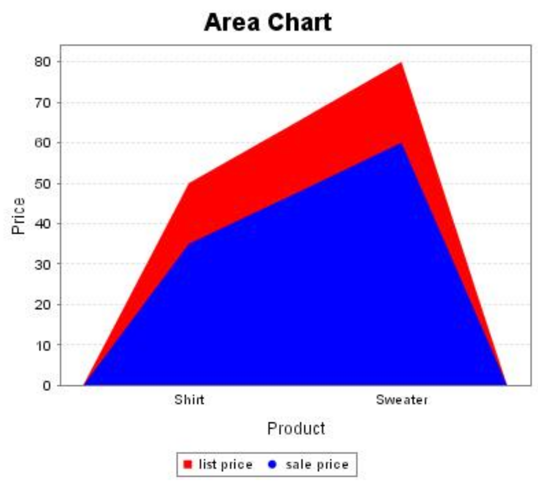
The following figures show an example of the **areaChart** type.

**Figure 128** Source Data for areaChart Example



```
PRODUCT_NAME         LIST_PRICE SALE_PRICE
-------------------- ---------- ----------
Shirt                        50         35
Sweater                      80         50
```

**Figure 129** areaChart Example



The **xyAreaChart** type displays (x,y) pairs of data. Rather than presenting a category axis and a value axis, this chart type presents two value axes.

## Bar

A bar chart displays the data points as vertical rectangles. eVision provides the following bar chart types:

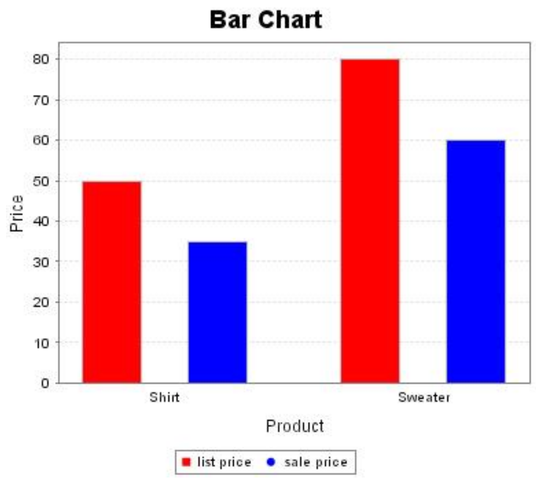
- **barChart**
- **barChart3D**
- **stackedBarChart**
- **stackedBarChart3D**

The following figures show an example of the **barChart** type.

**Figure 130** Source Data for barChart Example

```
PRODUCT_NAME         LIST_PRICE SALE_PRICE
-------------------- ---------- ----------
Shirt                        50         35
Sweater                      80         50
```

**Figure 131** barChart Example



The **barChart3D** and **stackedBarChart3D** types provide a 3-D visual effect.

The **stackedBarChart** and **stackedBarChart3D** types use a single bar to display the values for each category.

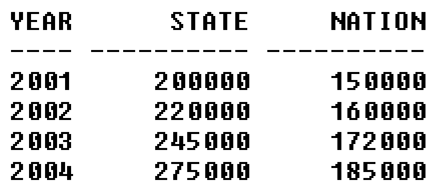
## Line

A line chart connects the data points with lines. eVision provides the following line chart types:

- **lineChart**
- **xyLineChart**

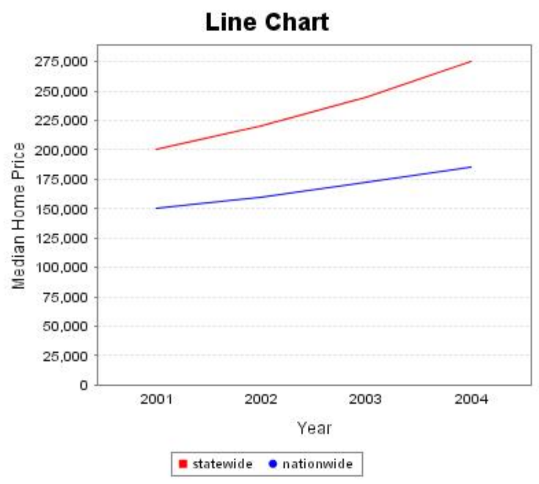
The following figures show an example of the **lineChart** type.

**Figure 132** Source Data for lineChart Example



```
YEAR      STATE      NATION
---- ---------- ----------
2001     200000     150000
2002     220000     160000
2003     245000     172000
2004     275000     185000
```

**Figure 133** lineChart Example



The **xyLineChart** type displays (x,y) pairs of data. Rather than presenting a category axis and a value axis, this chart type presents two value axes.

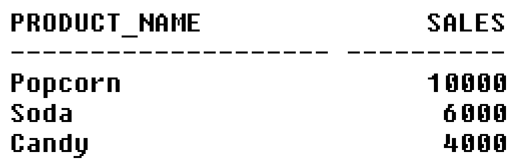
## Pie

A pie chart displays the data as a circle that has been divided into two or more wedge-shaped segments. eVision provides the following pie chart types:

- **pieChart**
- **pieChart3D**

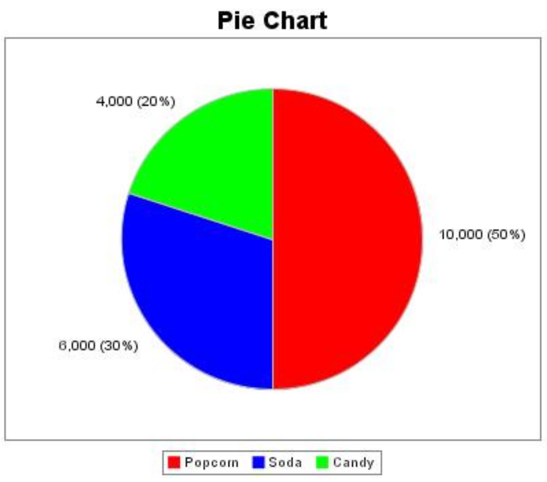
The following figures show an example of the **pieChart** type. In the chart, each segment is labeled with the raw number and the percentage.

**Figure 134** Source Data for pieChart Example



```
PRODUCT_NAME                  SALES
--------------------- ----------
Popcorn                       10000
Soda                           6000
Candy                          4000
```

**Figure 135** pieChart Example



The **pieChart3D** type provides a 3-D visual effect.

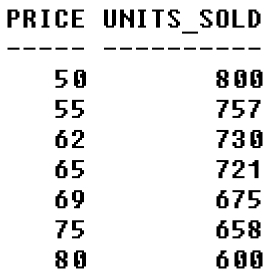
## Scatter Plot

A scatter plot chart illustrates the correlation between (x,y) pairs of data. Rather than presenting a category axis and a value axis, this chart type presents two value axes. Each data point is represented by a dot. eVision provides the following scatter plot type:

- **scatterPlotChart**

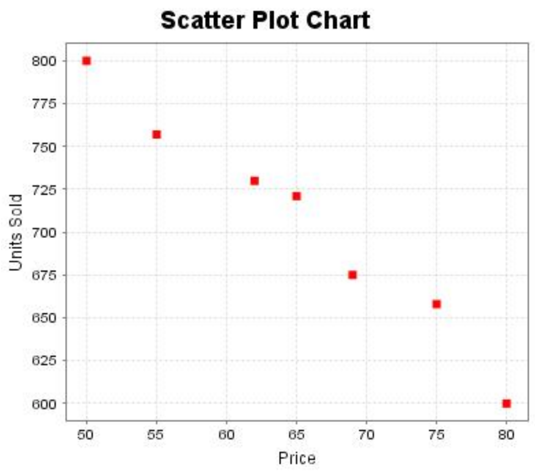
The following figures show an example of the **scatterPlotChart** type. In this example, the data has a high negative correlation.

**Figure 136** Source Data for scatterPlotChart Example



```
PRICE UNITS_SOLD
----- ----------
   50        800
   55        757
   62        730
   65        721
   69        675
   75        658
   80        600
```

**Figure 137** scatterPlotChart Example



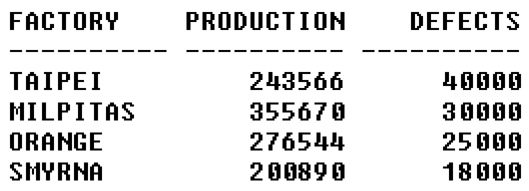
## Waterfall

A waterfall chart is a variation of the bar chart type. It displays the bars as floating vertical rectangles. eVision provides the following waterfall chart type:

- **waterfallChart**

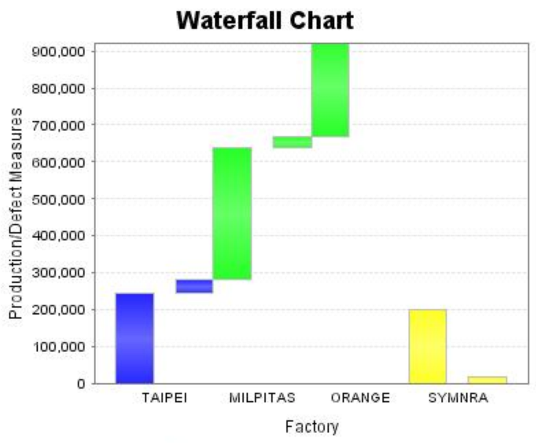
The following figures show an example of the **waterfallChart** type.

**Figure 138** Source Data for waterfallChart Example



```
FACTORY    PRODUCTION    DEFECTS
---------- ---------- ----------
TAIPEI         243566      40000
MILPITAS       355670      30000
ORANGE         276544      25000
SMYRNA         200890      18000
```

**Figure 139** waterfallChart Example



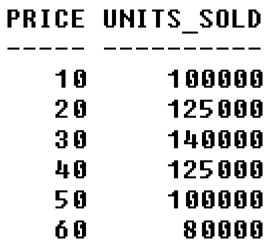
## XY Step Area

An xy step area chart illustrates the correlation between (x,y) pairs of data. Rather than presenting a category axis and a value axis, this chart type presents two value axes. Each data point is represented by a vertical rectangle. The rectangles are joined in a way that makes them resemble a series of steps. eVision provides the following xy step area chart type:

- **xyStepAreaChart**

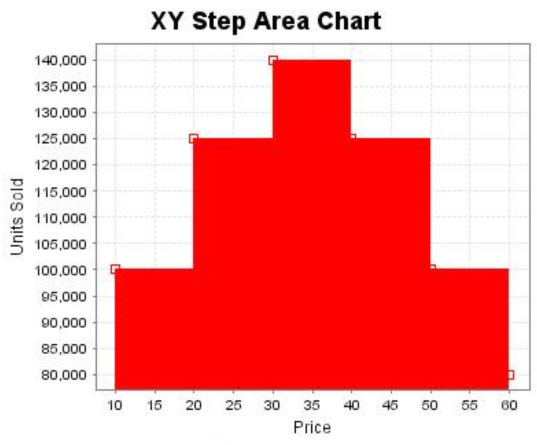
The following figures show an example of the **xyStepAreaChart** type.

**Figure 140** Source Data for xyStepAreaChart Example



```
PRICE UNITS_SOLD
----- ----------
   10     100000
   20     125000
   30     140000
   40     125000
   50     100000
   60      80000
```

**Figure 141** xyStepAreaChart Example

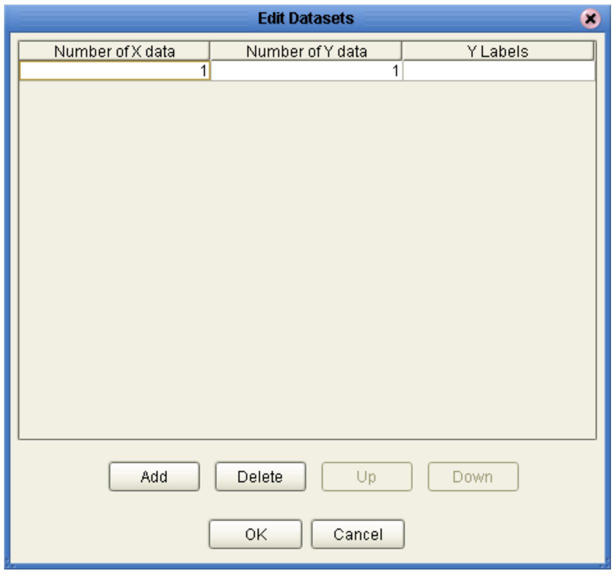


### 8.2.2 Data Sets

A data set is a group of related data points. Each chart has one or more data sets.

The **datasets** property enables you to specify the characteristics of each data set. When you click the right column, the **Edit Datasets** dialog box appears. See Figure 142.

**Figure 142** Edit Datasets Dialog Box



| Number of X data | Number of Y data | Y Labels |
|---|---|---|
| 1 | 1 | |

By default, the dialog box contains one data set.

For the chart types that are based on categories, you typically need to specify only one data set. For the chart types that are based on xy series, you may need to specify more than one data set. Table 15 lists the number of data sets for each chart type.

**Table 15** Chart Types and Number of Data Sets

| Chart Type | Based On | Number of Data Sets |
|---|---|---|
| areaChart | Categories | one |
| stackedAreaChart | Categories | one |
| xyAreaChart | XY Series | one or more |
| barChart | Categories | one |
| barChart3D | Categories | one |
| stackedBarChart | Categories | one |
| stackedBarChart3D | Categories | one |
| lineChart | Categories | one |
| xyLineChart | XY Series | one or more |
| pieChart | Categories | one |
| pieChart3D | Categories | one |
| scatterPlotChart | XY Series | one or more |

**Table 15** Chart Types and Number of Data Sets

| Chart Type | Based On | Number of Data Sets |
|---|---|---|
| waterfallChart | Categories | one |
| xyStepAreaChart | XY Series | one or more |

You will map data into the data set elements in the Page Flow Designer. See **"Mapping Data into the Chart" on page 153**.

### Guidelines for Category Series Charts

The **Number of X data** column specifies the number of input sources to be used for the categories. For example, if you were creating the bar chart in **Figure 131 on page 145**, you would set this column to 1.

The **Number of Y data** column specifies the number of values for each category. For example, if you were creating the bar chart in **Figure 131 on page 145**, you would set this column to 2.

If you are creating a pie chart, then you must set the **Number of Y data** column to 1.

The **Y Labels** column enables you to specify the names in the chart legend. Separate the values with a comma (as in **list price,sale price**).

### Guidelines for XY Series Charts

The **Number of X data** column specifies the number of input sources to be used for the x portion of the (x,y) pairs. For example, if you were creating the scatter plot chart in **Figure 137 on page 148**, you would set this column to 1.

The **Number of Y data** column specifies the number of input sources to be used for the y portion of the (x,y) pairs. For example, if you were creating the scatter plot chart in **Figure 137 on page 148**, you would set this column to 1.

The **Y Labels** column enables you to specify the names in the chart legend (as in **Price,Units Sold**).

If the number of (x,y) pairs in each series is different, you must specify more than one data set.

## 8.2.3 Additional Properties

As with all of the eVision Studio GUI components, be sure to change the default value of the **lname** property to a more descriptive value.

The **bgColor** property indicates the color that appears in the background of the chart. The default color is white.

You can resize a chart using the mouse, or by changing values of the **height** and **width** properties.

The **legend** property indicates the location of the chart legend: north, east, south, or west. You can also remove the chart legend.

The **orientation** property indicates whether the chart orientation is vertical or horizontal. The default setting is vertical.

***Note:*** *The pie chart types do not use the* ***orientation*** *property.*

The **title** property enables you to add a title to the top of the chart. In addition, the value appears as a tooltip.

The **xAxisLabel** property enables you to specify a label for the category axis.

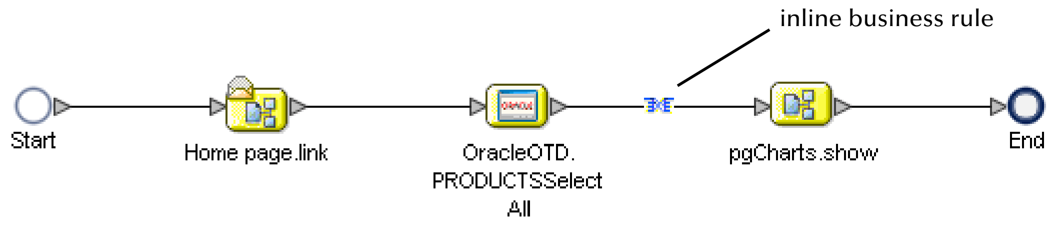
The **yAxisLabel** property enables you to specify a label for the value axis.

## 8.3 Mapping Data into the Chart

After adding the chart to a Page Layout, you specify what data will appear in the chart. You perform this step in the Page Flow Designer.

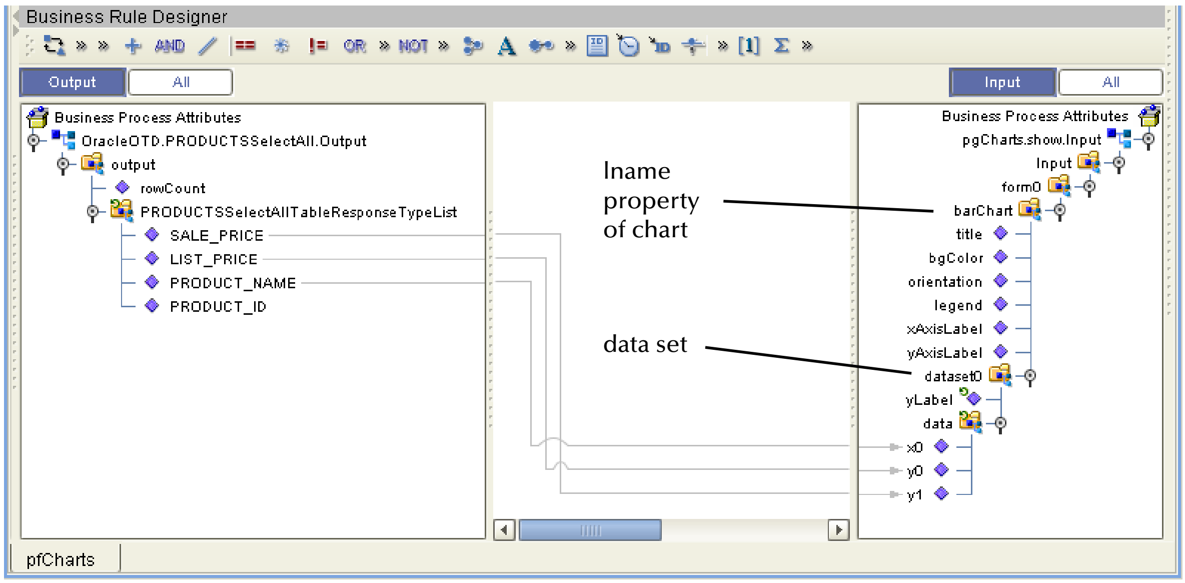
The example in this section maps data from an Oracle database into the chart. Figure 143 shows the Page Flow.

**Figure 143** Page Flow for Mapping Example



The mapping occurs in the inline business rule between the **SelectAll** operation of the Oracle OTD and the **show** operation of the Page Layout. Figure 144 shows the Business Rule Designer view of the inline business rule.

**Figure 144** Inline Business Rule Mapping



The Output pane (on the left) represents the output from the **SelectAll** operation of the Oracle OTD. The OTD was created for a table called **PRODUCTS** that contains the following columns: **PRODUCT_ID**, **PRODUCT_NAME**, **LIST_PRICE**, and **SALE_PRICE**.

The Input pane (on the right) represents the input to the Page Layout. Each chart in the Page Layout is identified by its **lname** property. Figure 144 shows one chart named **barChart**.

The data set in each chart is identified by **dataset*n***. The numeric portion starts from 0.

The x and y nodes under each data set correspond to the values that you entered in the **Edit Datasets** dialog box. The numeric portion starts from 0. For example, if you entered 2 in the **Number of X data** column and 4 in the **Number of Y data** column, the following nodes would appear:

- x0
- x1
- y0
- y1
- y2
- y3

In the example, the **PRODUCT_NAME** column represents the categories that will appear in the bar chart. Therefore, this column is mapped to the **x0** node.

The **LIST_PRICE** and **SALE_PRICE** columns represent the values that will appear for each category. Therefore, these columns are mapped to the **y0** and **y1** nodes, respectively.

**Figure 131 on page 145** shows how the chart appears when the Project is activated.

Chapter 9

# Authentication and Error Handling

This chapter describes how to add authentication to eVision Web applications, as well as how to return preconfigured pages for certain errors.

**This Chapter Includes:**

## 9.1 Overview

You have for three authentication options for an eVision Web application:

- No authentication.
- Using the default ICAN authentication.
- Using the preconfigured authentication pages that are provided with eVision. This is the most powerful option.

Table 16 describes the preconfigured authentication pages.

**Table 16** Preconfigued Authentication Pages

| Authentication Page | Description | Required? |
|---|---|---|
| Login Page | This page allows the user to enter login information. | yes |
| Login Error | This page is returned if the user enters an invalid username and/or password. | yes |
| Access Denied Error | This page is returned if the user is not allowed to see the requested eVision page. Access Denied Error corresponds to the HTTP response codes 401 and 403. These response codes are two variations of the same error. | no |

You can also return preconfigured error-handling pages to the user when a requested page cannot be found or when a more serious internal error has occurred. You can create these pages regardless of which authentication option you are using.

Table 17 describes the preconfigured error-handling pages that are provided with eVision.

**Table 17** Preconfigured Error-Handling Pages

| Authentication Page | Description |
|---|---|
| No Such Resource | This page is returned if the requested eVision page cannot be found. No Such Resource corresponds to the HTTP response code 404. |
| Internal Server Error | This page is returned to the user if there is a serious problem with the business process (for example, the business process is corrupt or the session was lost). Internal Server Error corresponds to the HTTP response code 500. |

## 9.2 Creating Authentication and Error-Handling Pages

The following procedure describes how to add a preconfigured authentication page or a preconfigured error-handling page to your Project. Perform this procedure for each page that you want to create.

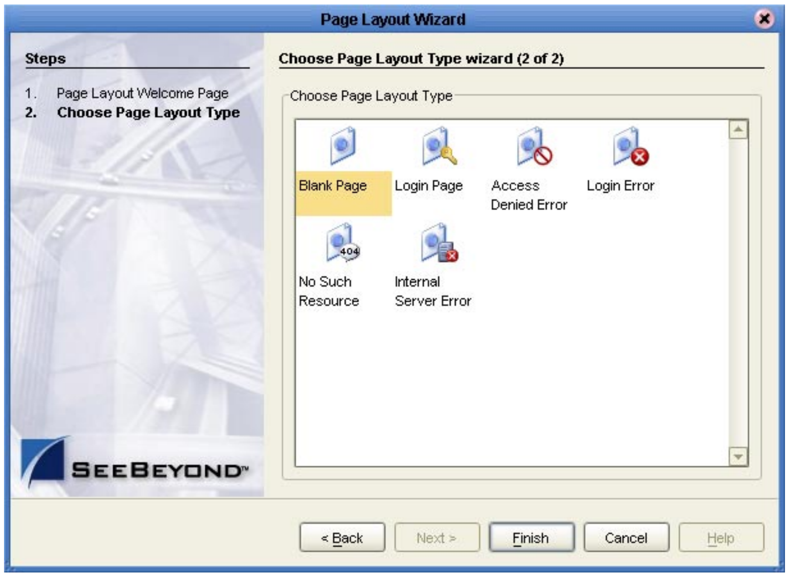
**To create an authentication or error-handling page**

1. In the Project Explorer of Enterprise Designer, right-click the Project.
2. On the context menu, click **New**, and then select **Page Layout**.

   Step 1 of the Page Layout Wizard appears.
3. In the **Page Layout Name** field, type a unique name for the new Page Layout (for example, **MyLoginPage**).
4. Click **Next**.

   Step 2 of the Page Layout Wizard appears.

**Figure 145** Page Layout Wizard Page 2



5 Select the authentication or error-handling page that you want to create.

6 Click **Finish**.

The Page Layout Designer appears with the page that you chose.

7 Because the page is preconfigured, you do not need make any changes in the Page Layout Designer. However, you might want to add a background or modify the title. You can perform these tasks in the **Properties** window.

## 9.3 Configuring the Connectivity Map

If you are using the default ICAN authentication or the preconfigured authentication pages, you must set configuration parameters for the Web Connector in the Connectivity Map.

You must also set configuration parameters for the Web Connector if you are using either of the preconfigured error-handling pages.

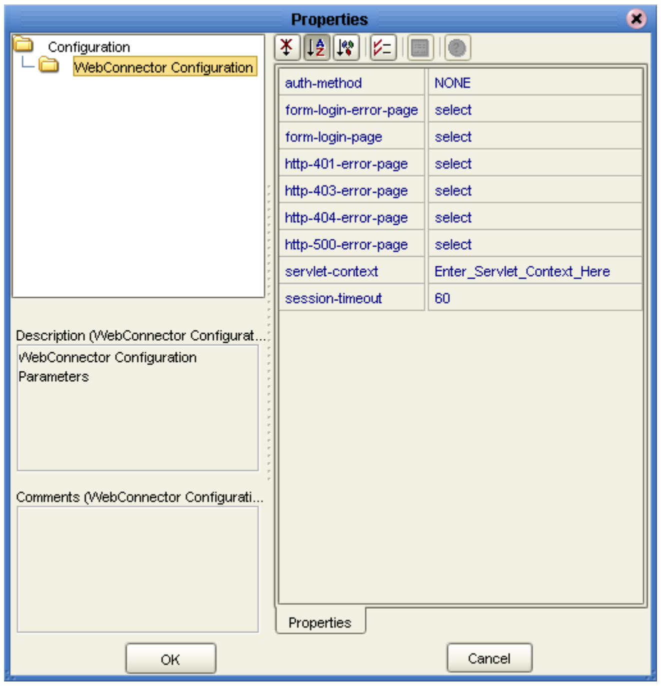
The following procedure assumes that you have created the Connectivity Map for your application. **"Creating the Connectivity Map" on page 136** describes how to do this.

**To configure authentication in the Connectivity Map**

1 In the Project Explorer of Enterprise Designer, right-click the Connectivity Map and select **Open**. The Connectivity Map Editor appears.

2 Open the service binding box and double-click the connection icon between the **WSPProvider** Implemented Service and the Web Connector. The **Properties** dialog box appears. See Figure 146.

**Figure 146** WebConnector Configuration Properties


Properties
Configuration
WebConnector Configuration
Description (WebConnector Configurat...
Comments (WebConnector Configurati...
Properties
OK
Cancel

WebConnector Configuration Parameters

| | |
|---|---|
| auth-method | NONE |
| form-login-error-page | select |
| form-login-page | select |
| http-401-error-page | select |
| http-403-error-page | select |
| http-404-error-page | select |
| http-500-error-page | select |
| servlet-context | Enter_Servlet_Context_Here |
| session-timeout | 60 |

3 If you want to use the default ICAN authentication, set the **auth-method** property to **BASIC**.

4 If you want to use the preconfigured authentication pages, do the following:

A Set the **auth-method** property to **FORM**.

B Set the **form-login-error-page** property to the Login Error Page that you created.

C Set the **form-login-page** property to the Login Page that you created.

D If you created an Access Denied Error Page, set the **http-401-error-page** and **http-403-error-page** properties to the Access Denied Error Page.

***Note:*** *The **Properties** dialog box enables you to clear the values of the page properties.*

5 If you created a No Such Resource Page, set the **http-404-error-page** property to the No Such Resource Page.

6 If you created an Internal Server Error Page, set the **http-500-error-page** property to the Internal Server Error Page.

7 Click **OK**.

## 9.4 Specifying Users and Roles

The final step of implementing the default ICAN authentication or the preconfigured authentication pages is to specify which users and roles can access the application.

This step involves adding users and roles to the Environment and modifying the eVision External System to indicate which roles are authorized.

The following procedures assume that you have created an Environment for your application. The Environment must include an eVision External System. For more information, see **"Creating the Environment" on page 137**.

### To add users and roles to the Environment

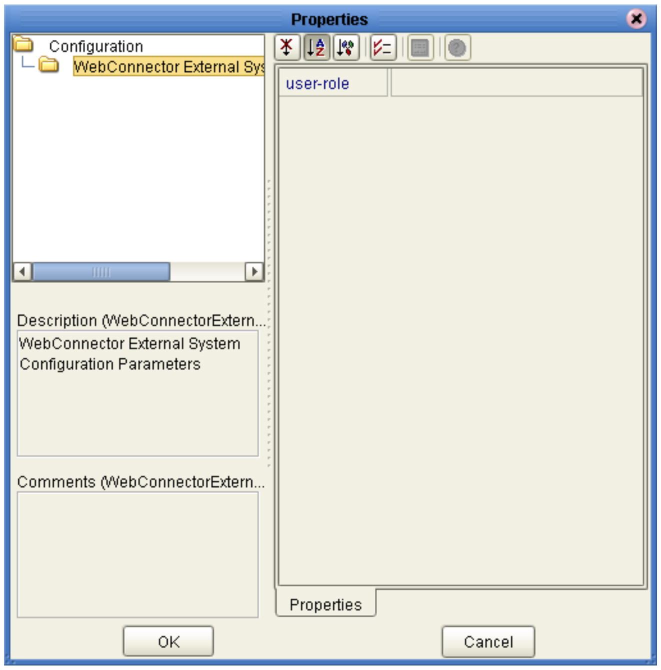
The "ICAN Security Features" chapter in the *eGate Integrator System Administration Guide* contains detailed information about how to add users and roles to an Environment. The following procedure is a condensed version of this information.

1. In the Environment Explorer of Enterprise Designer, right-click the Environment and select **User Management**. The **User Management** dialog box appears.
2. Click **Add**.
3. In the **User** field, enter a name for the user.
4. In the **Password** field, enter a password for the user.
5. In the **Confirm Password** field, enter the password again.
6. Click **Add Role**.
7. If you want to create a new role that can be assigned, do the following:
   A. Click **Create Role**.
   B. In the **Role** field, type the name of the role.
   C. Click **OK**.
8. Select the desired role(s) and click **OK**.
9. Click **OK**.
10. Click **Close**.

## To modify the eVision External System

1 In the Environment Explorer of Enterprise Designer, right-click the eVision External System and select **Properties**. The **Properties** dialog box appears. See Figure 147.

**Figure 147** eVision External System Properties


Properties
Configuration
WebConnector External Sys
user-role
Description (WebConnectorExtern...
Comments (WebConnectorExtern...
Properties
OK
Cancel

WebConnector External System Configuration Parameters

2 Select the **user-role** property and click the Command button (**...**).

3 For each role that you want to have access to the application, do the following:

A Click **Add**. The **Input** dialog box appears.

B Enter the role name (for example, **analyst**).

C Click **OK**.

4 When you are done adding roles, click **OK** to return to the **Properties** dialog box.

5 Click **OK**.

Appendix A

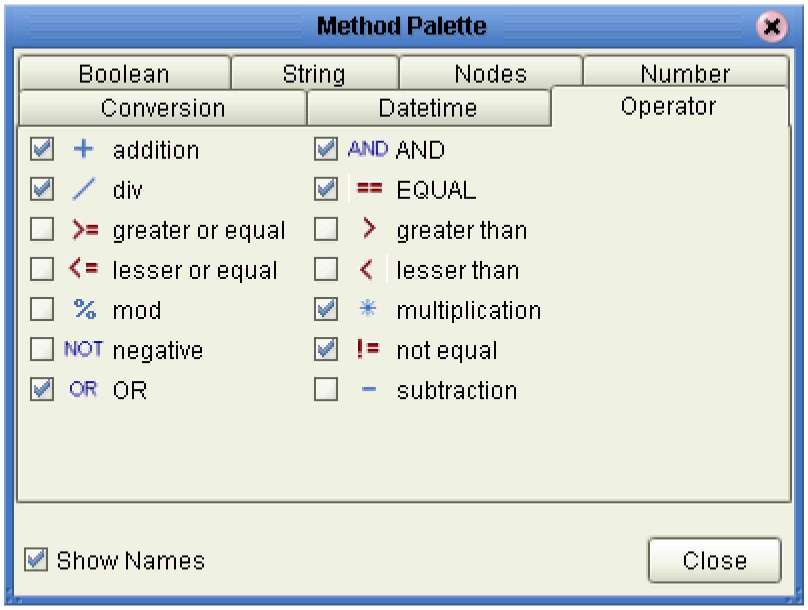
# Method Palette

This appendix describes each method that appears in the Method Palette of the Business Rule Designer.

## A.1 Operators

Operators are the methods that allow you to manipulate data with standard mathematical operators.

**Figure 148** Method Palette: Operator Tab



| | | | | | |
|---|---|---|---|---|---|
| ✓ | + | addition | ✓ | AND | AND |
| ✓ | / | div | ✓ | == | EQUAL |
| ☐ | >= | greater or equal | ☐ | > | greater than |
| ☐ | <= | lesser or equal | ☐ | < | lesser than |
| ☐ | % | mod | ✓ | * | multiplication |
| ☐ | NOT | negative | ✓ | != | not equal |
| ✓ | OR | OR | ☐ | - | subtraction |

✓ Show Names

**Table 18** Operator Methods

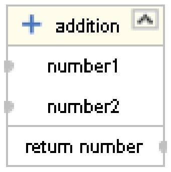
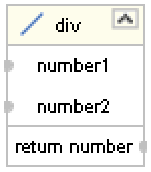
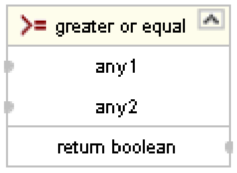
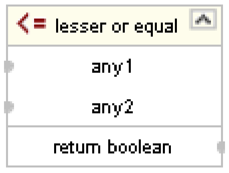
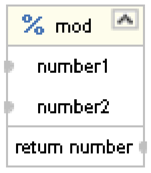
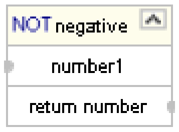
| Symbol | Name | Function |
|---|---|---|
|  | addition | Adds the value of *number1* to the value of *number2*, returns the sum. |
|  | div | Divides the value of *number1* by the value of *number2*, returns the quotient. |
|  | greater or equal | Returns Boolean true if *number1* is greater than or equal to *number2*; otherwise, returns Boolean false. |
|  | lesser or equal | Returns Boolean true if *number1* is less than or equal to *number2*; otherwise, returns Boolean false. |
|  | mod | Used to divide two numbers and return only the remainder. |
|  | negative | Converts the input number to negative. Result is a negative number having the same absolute value as the input number. |

**Table 18** Operator Methods (Continued)

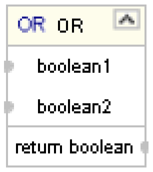
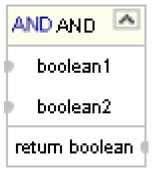
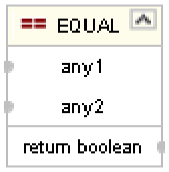
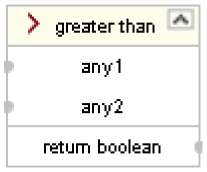
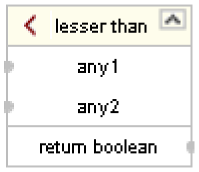
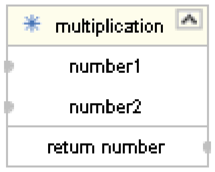
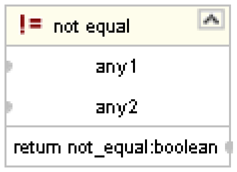
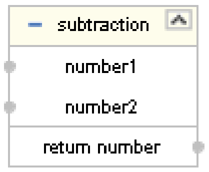
| Symbol | Name | Function |
|---|---|---|
|  OR OR boolean1 boolean2 return boolean | OR | Returns Boolean false if both *boolean1* and *boolean2* are false; otherwise, returns Boolean true. |
|  AND AND boolean1 boolean2 return boolean | AND | Returns Boolean true if both *boolean1* and *boolean2* are true; otherwise, returns Boolean false. |
|  EQUAL any1 any2 return boolean | EQUAL | Returns Boolean true if *number1* is equal to *number2*; otherwise, returns Boolean false. |
|  greater than any1 any2 return boolean | greater than | Returns Boolean true if *number1* is greater than *number2*; otherwise, returns Boolean false. |
|  lesser than any1 any2 return boolean | lesser than | Returns Boolean true if *number1* is less than *number2*; otherwise, returns Boolean false. |
|  multiplication number1 number2 return number | multiplication | Multiplies the value of *number1* by the value of *number2*, returns the product. |

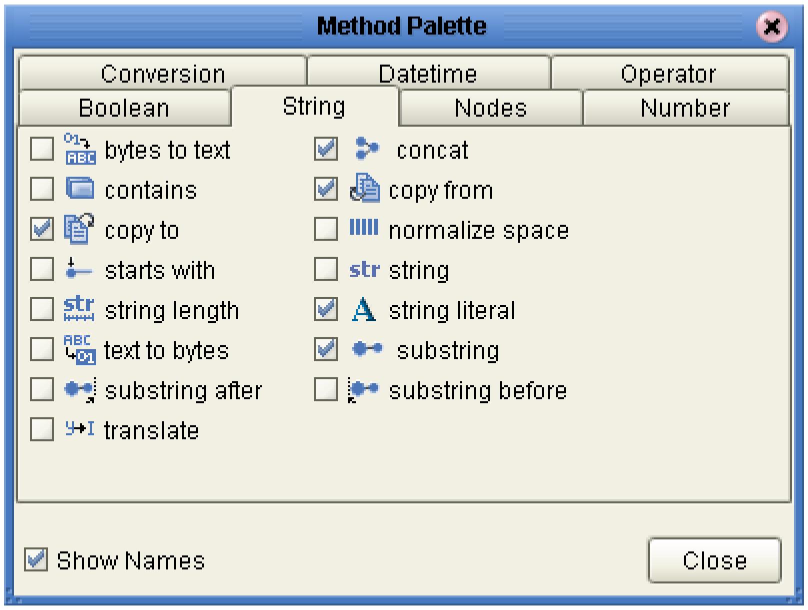
**Table 18** Operator Methods (Continued)

| Symbol | Name | Function |
|---|---|---|
|  | not equal | Returns Boolean true if *number1* is not equal to *number2*; otherwise, returns Boolean false. |
|  | subtraction | Subtracts the numerical value of *number2* from the numerical value of *number1*, returns the difference. |

## A.1 String

The String methods allow you to manipulate string data.

**Figure 149** Method Palette: String Tab



Method Palette

Tabs: Conversion, Datetime, Operator, Boolean, String, Nodes, Number

| | |
|---|---|
| ☐ bytes to text | ☑ concat |
| ☐ contains | ☑ copy from |
| ☑ copy to | ☐ normalize space |
| ☐ starts with | ☐ string |
| ☐ string length | ☑ string literal |
| ☐ text to bytes | ☑ substring |
| ☐ substring after | ☐ substring before |
| ☐ translate | |

☑ Show Names

Close

**Table 19** String Methods

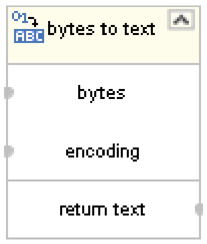
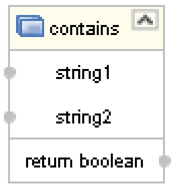
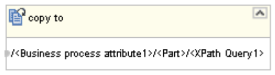
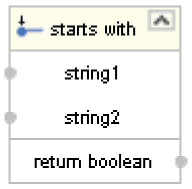
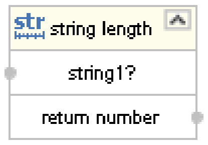
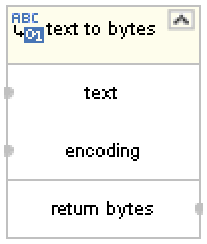
| Symbol | Name | Function |
|---|---|---|
|  | bytes to text | Decodes bytes into text using the specified encoding. If no encoding is specified, the platform's default encoding is used. |
|  | contains | Returns true if the second string is contained within the first string, otherwise it returns false |
|  | copy to | Allows you to type in the xpath expression for the destination of a copy operation. This is useful for entering xpath predicates. Note: This is for advanced users who are familiar with xpath and BPEL syntax. |
|  | starts with | Returns true if the first string starts with the second string, otherwise it returns false |
|  | string length | Returns the number of characters in a string |
|  | text to bytes | Encodes the input text into a sequence of bytes using the specified encoding. If no encoding is specified, the platform's default encoding is used |

**Table 19** String Methods (Continued)

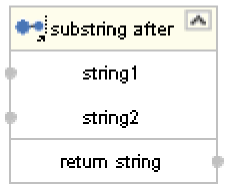
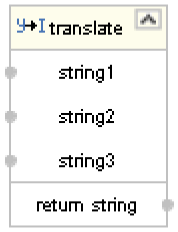
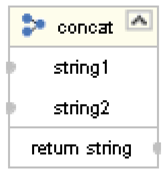
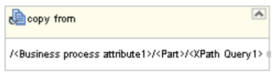
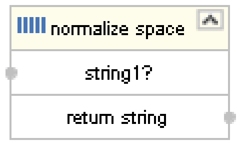
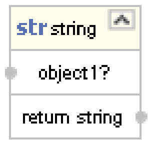
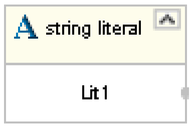
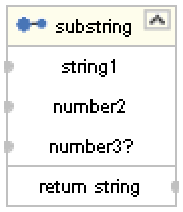
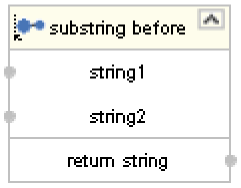
| Symbol | Name | Function |
|---|---|---|
|  | substring after | Returns the part of the string in the string argument that occurs after the substring in the substring argument |
|  | translate | Performs a character by character replacement. It looks in the value argument for characters contained in string1, and replaces each character for the one in the same position in the string2 |
|  | concat | Returns the concatenation of all its arguments |
|  copy from<br>/&lt;Business process attribute1&gt;/&lt;Part&gt;/&lt;XPath Query1&gt; | copy from | Allows you to type in xpath expression for the source of a copy operation. This is useful for entering xpath predicates. Note: This is for advanced users who are familiar with xpath and BPEL syntax |
|  | normalize space | Removes leading and trailing spaces from a string |
|  | string | Converts the value argument to a string |

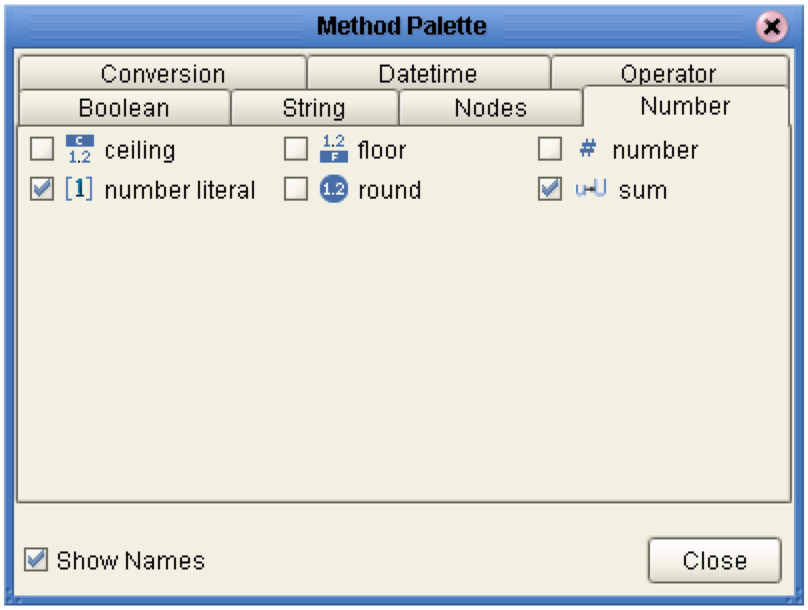
**Table 19** String Methods (Continued)

| Symbol | Name | Function |
|---|---|---|
|  | string literal | A sequence of characters of fixed length and content |
|  substring<br>string1<br>number2<br>number3?<br>return string | substring | Returns a part of the string in the string argument |
|  | substring before | Returns the part of the string in the string argument that occurs before the substring in the substring argument. |

## A.2 Number

The Number methods allow you to work with number data.

**Figure 150** Method Palette: Number Tab



- [ ] ceiling
- [ ] floor
- [ ] number
- [x] number literal
- [ ] round
- [x] sum
- [x] Show Names

**Table 20** Number Methods

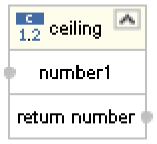
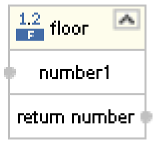
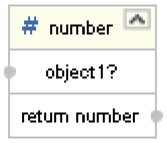
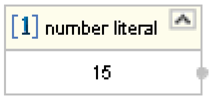
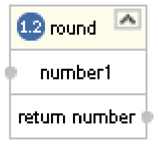
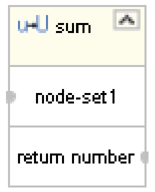
| Symbol | Name | Function |
|---|---|---|
|  | ceiling | Returns the smallest integer that is not less than the number argument |
|  | floor | Returns the largest integer that is not greater than the number argument |
|  | number | Converts the value argument to a number |
|  | number literal | A literal number string of fixed length and content |

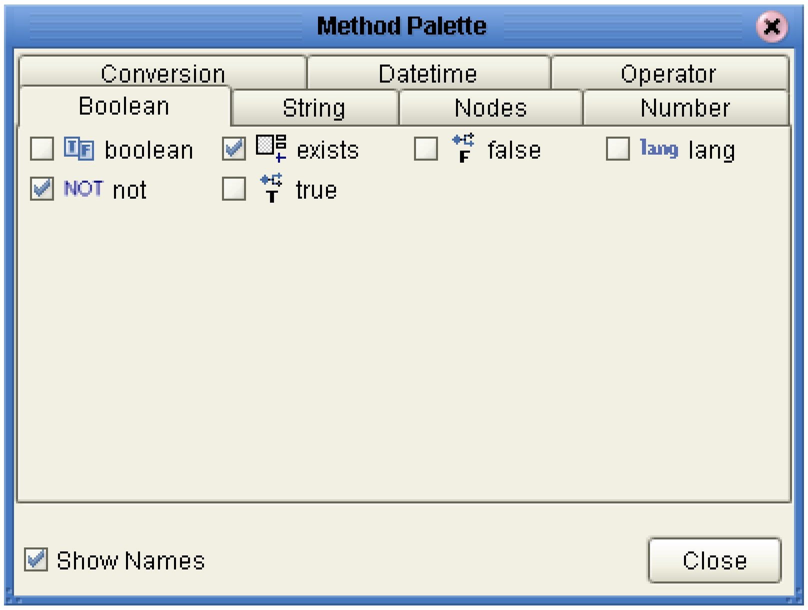
**Table 20** Number Methods (Continued)

| Symbol | Name | Function |
|---|---|---|
|  round / number1 / return number | round | Rounds the number argument to the nearest integer |
|  sum / node-set1 / return number | sum | Returns the total value of a set of numeric values in a node-set |

## A.3 Boolean

Boolean methods allow you to apply boolean logic to your data.

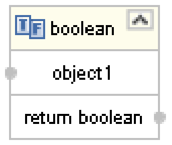
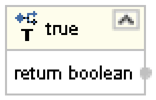
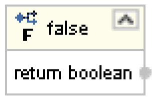
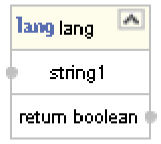
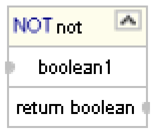
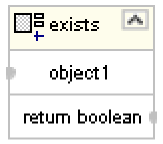
**Figure 151** Method Palette: Boolean Tab



☐ boolean ☑ exists ☐ false ☐ lang
☑ not ☐ true

☑ Show Names

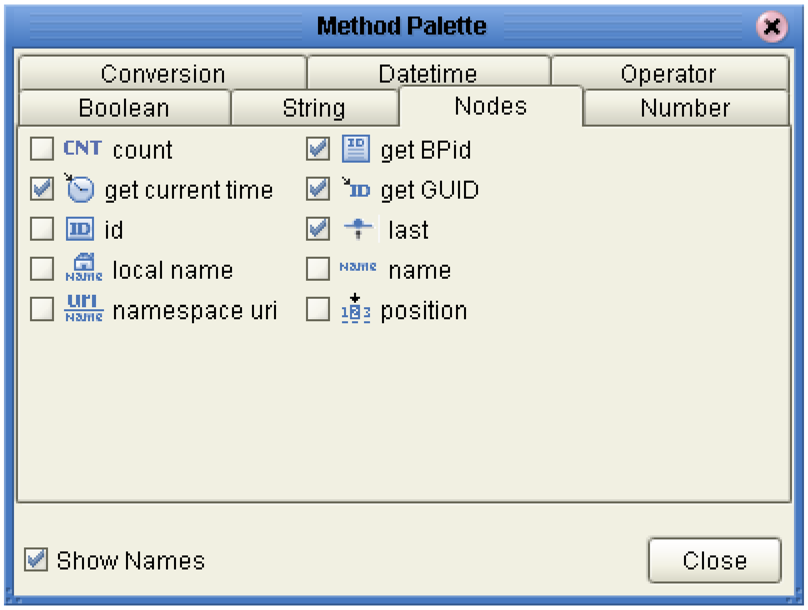
**Table 21** Boolean Methods

| Symbol | Name | Function |
|---|---|---|
|  | boolean | Converts the value argument to Boolean and returns true or false. |
|  | true | Returns true |
|  | false | Returns false |
|  | lang | Returns true if the language argument matches the language of the xsl:lang element, otherwise it returns false. |
|  | not | Returns true if the condition argument is false, and false is the condition argument is true. |
|  | exists | Checks to see if a value is present and returns a Boolean result. |

## A.4 Nodes

Node methods allow you to manipulate your data.

**Figure 152** Method Palette: Nodes Tab



☐ count
✓ get current time
☐ id
☐ local name
☐ namespace uri
✓ get BPid
✓ get GUID
✓ last
☐ name
☐ position
✓ Show Names

**Table 22** Nodes Methods

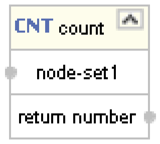
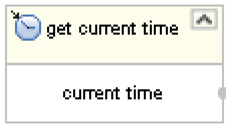
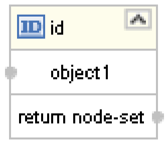
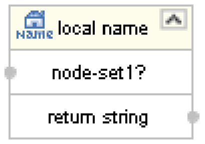
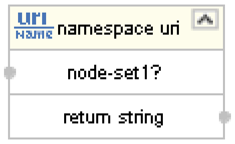
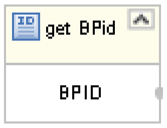
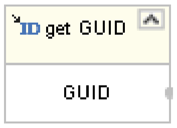
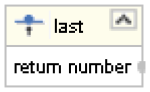
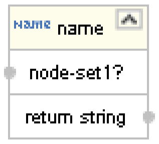
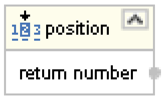
| Symbol | Name | Function |
|---|---|---|
|  | count | Returns the number of nodes in a node-set |
|  | get current time | Gets the current time in ISO 8601 format (e.g. 2003-08-15T02:03:49.92Z). |
|  | id | Selects elements by their unique ID |
|  | local name | Returns the local part of a node. A node usually consists of a prefix, a colon, followed by the local name |

**Table 22** Nodes Methods (Continued)

| Symbol | Name | Function |
|---|---|---|
|  | namespace uri | Returns the namespace URI of a specified node |
|  | get BPid | Gets the business process instance ID. |
|  | get GUID | Gets a randomly generated globally unique ID. |
|  | last | Returns the position number of the last node in the processed node list |
|  | name | Returns the name of a node |
|  | position | Returns the position in the node list of the node that is currently being processed |

## A.5 Datetime

Datetime methods allow you to manipulate date, time, and duration of data.

**Figure 153** Method Palette: Datetime Tab

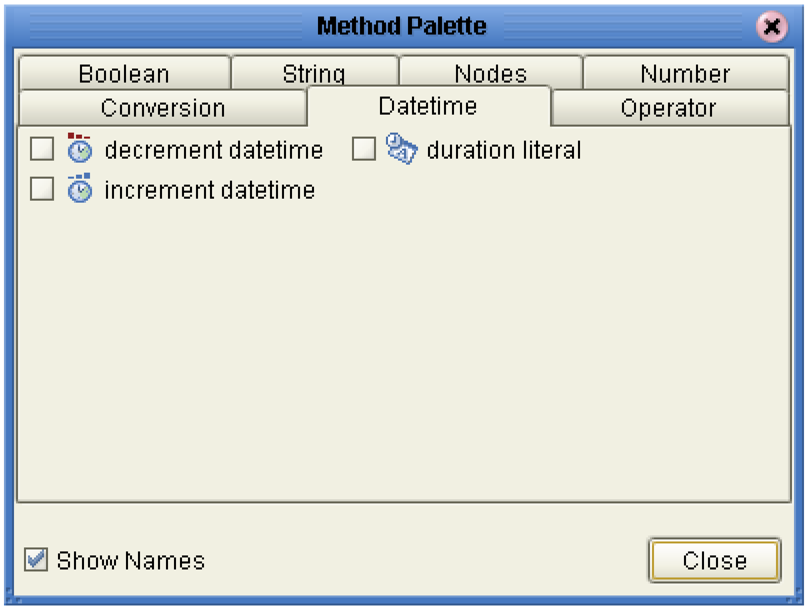
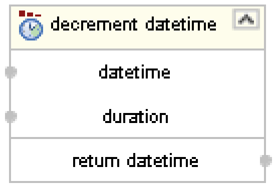
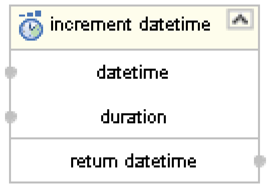
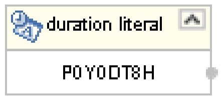


**Table 23** Datetime Methods

| Symbol | Name | Function |
|---|---|---|
|  decrement datetime<br>datetime<br>duration<br>return datetime | decrement datetime | Dynamically decreases the date or time by a certain duration, such as days or hours. |
|  increment datetime<br>datetime<br>duration<br>return datetime | increment datetime | Dynamically increases the date or time by a certain duration, such as days or hours. |
|  | duration literal | Allows you to set an actual date or time. |

## A.6 Conversion

The Convert method allows you to make conversions from various data types.

**Figure 154** Method Palette: Conversion Tab

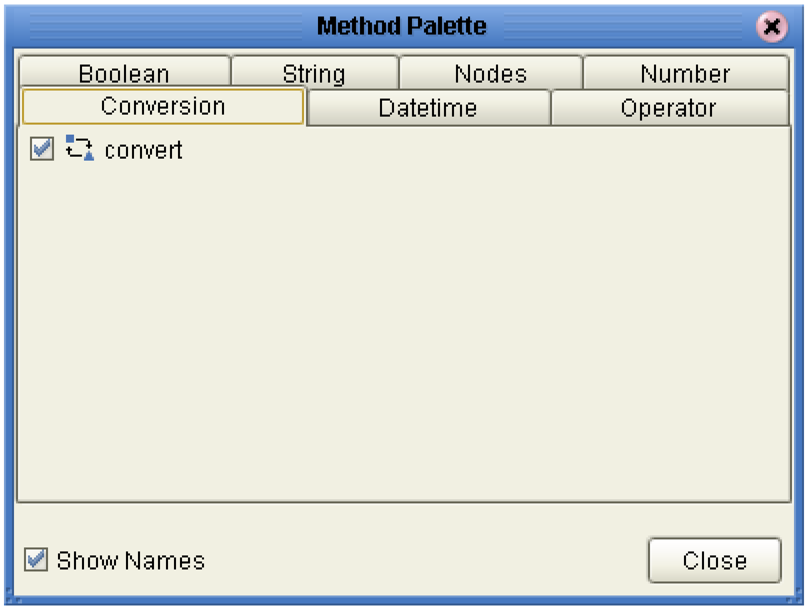
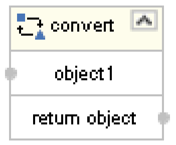


**Table 24** Conversion Methods

| Symbol | Name | Function |
|---|---|---|
|  | convert | The convert function that takes in one input link and one output link. The data type conversions are described in **"Data Type Conversions" on page 174**. |

### A.6.1. Data Type Conversions

The Business Rule Designer supports a Convert function that takes in one input link and one output link. The Convert function is implemented from tree to tree mapping only. The Convert function is valid for conversions between leaf nodes. The Conversion function checks if the mapping is valid. The valid conversions are based off the following conversions.

## String

**Table 25 String**

| To | From |
|---|---|
| Boolean | custom |
| Float | parse |
| Double | parse |
| Decimal | parse |
| Byte | parse |
| Short | parse |
| Int | parse |
| Long | parse |
| Duration | parse |
| dateTime | parse |
| time | parse |
| date | parse |
| gYearMonth | parse |
| gYear | parse |
| gMonthDay | parse |
| gDay | parse |
| gMonth | parse |
| hexBinary | textToByte |
| base64Binary | textToByte |
| anyURI | parse |
| QName | parse |
| NOTATION | parse |

## Boolean

**Table 26 Boolean**

| To | From |
|---|---|
| String | toString |

## Float

**Table 27 Float**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| Double | floatToDouble |
| Decimal | floatToDecimal |
| Byte | floatToByte |
| Short | floatToShort |
| Int | floatToInt |
| Long | floatToLong |

## Double

**Table 28 Double**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| Float | doubleToFloat |
| Decimal | doubleToDecimal |
| Byte | doubleToByte |
| Short | doubleToShort |
| Int | doubleToInt |
| Long | doubleToLong |

## Decimal

**Table 29 Decimal**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| Float | decimalToFloat |
| Double | decimalToDouble |
| Byte | decimalToByte |
| Short | decimalToShort |

**Table 29 Decimal (Continued)**

| To | From |
|---|---|
| Int | decimalToInt |
| Long | decimalToLong |

## Byte

**Table 30 Byte**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean() |
| Float | byteToFloat |
| Double | byteToDouble |
| Decimal | byteToDecimal |
| Short | byteToShort |
| Int | byteToInt |
| Long | byteToLong |

## Short

**Table 31 Short**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean() |
| Float | shortToFloat |
| Double | shortToDouble |
| Decimal | shortToDecimal |
| Byte | shortToByte |
| Int | shortToInt |
| Long | shortToLong |

### Int

**Table 32 Int**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean() |
| Float | intToFloat |
| Double | intToDouble |
| Decimal | intToDecimal |
| Byte | intToByte |
| Short | intToShort |
| Long | intToLong |

### Long

**Table 33 Long**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean() |
| Float | longToFloat |
| Double | longToDouble |
| Decimal | longToDecimal |
| Byte | longToByte |
| Short | longToShort |
| Int | longToInt |

### Duration

**Table 34 Duration**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

### dateTime

**Table 35 dateTime**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| time | dateTimeToTime |
| date | dateTimeToDate |
| gYearMonth | dateTimeToGYearMonth |
| gYear | dateTimeToGYear |
| gMonthDay | dateTimeToGMonthDay |
| gDay | dateTimeToGDay |
| gMonth | dateTimeToGMonth |

### time

**Table 36 time**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

### date

**Table 37 date**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| gYearMonth | dateToGYearMonth |
| gYear | dateToGYear |
| gMonthDay | dateToGMonthDay |
| gDay | dateToGDay |
| gMonth | dateToGMonth |

### gYearMonth

**Table 38 gYearMonth**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| gYear | gYearMonthToGYear |
| gMonth | gYearMonthToGMonth |

### gYear

**Table 39 gYear**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

### gMonthDay

**Table 40 gMonthDay**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |
| gDay | gMonthDayToGDay |
| gMonth | gMonthDayToGMonth |

### gDay

**Table 41 gDay**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

### gMonth

Table 42 gMonth

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

### hexBinary

Table 43 hexBinary

| To | From |
|---|---|
| String | byteToText |
| Boolean | boolean |
| base64Binary | hexBinaryToBase64Binary |

### base64Binary

Table 44 base64Binary

| To | From |
|---|---|
| String | byteToText |
| Boolean | boolean |
| hexBinary | base64BinaryToHexBinary |

### anyURI

Table 45 anyURI

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

## QName

**Table 46 QName**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

## NOTATION

**Table 47 NOTATION**

| To | From |
|---|---|
| String | toString |
| Boolean | boolean |

Appendix B

# Conversational State in eVision Studio Web Applications

This appendix describes the support for conversational state in eVision Studio Web applications.

**This Chapter Includes:**

## B.1 Overview

HTTP is a stateless protocol, which means that Web applications must use some type of mechanism to maintain conversational state with clients. For example, the application might store a cookie on the user's computer.

In eVision Studio Web applications, Event Based Decision elements have the ability to maintain conversational state. When an Event Based Decision element is accessed in a Page Flow, the element keeps track of which Pages the user accesses. The user can leave a page and return to it as often as necessary.

This feature is particularly effective in promoting smooth page transitions when a user is moving from page to page in an unpredictable or non-sequential manner.

## B.2 Example

The following Page Flow will be used to illustrate conversational state in eVision Studio. Because of the size of the Page Flow, it is shown in two figures. The right boundary of Figure 155 continues to the left boundary of Figure 156. The letters A and B and the numbers 1 through 13 are used in the explanation that follows.

**Figure 155** Page Flow Example - Part 1

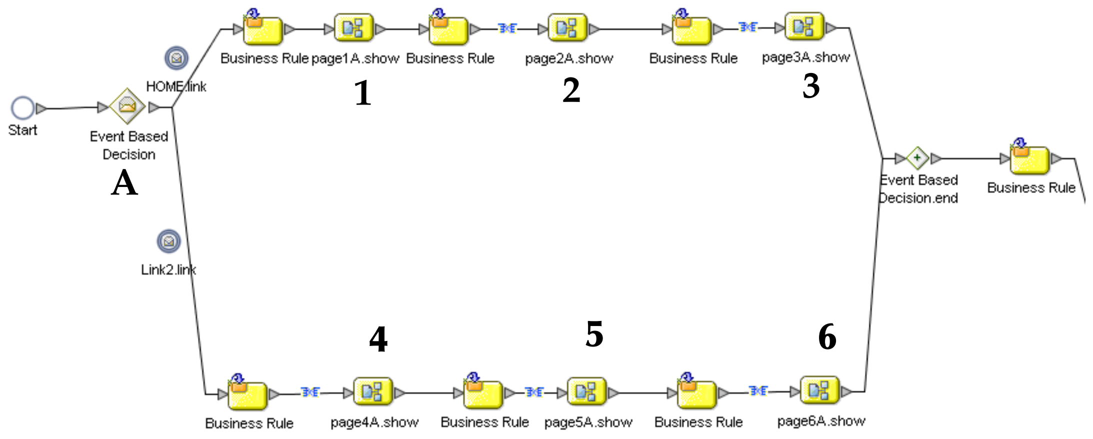
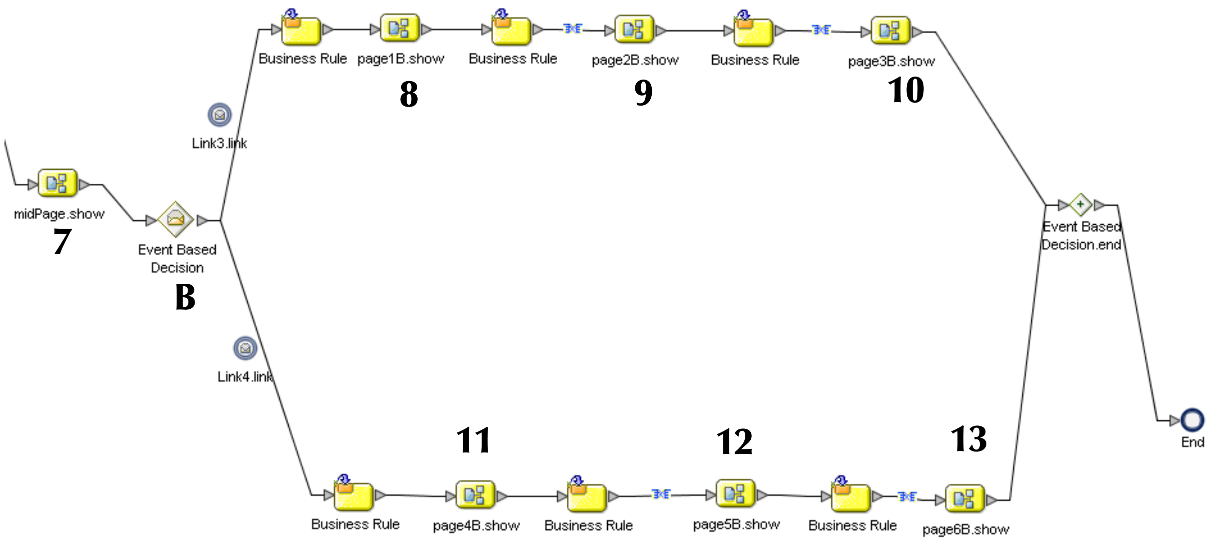


**Figure 156** Page Flow Example - Part 2



The Page Flow contains two Event Based Decision elements:

- Event Based Decision A contains six pages (1 through 6).
- Event Based Decision B contains six pages (8 through 13).

Page 7 is located between the Event Based Decisions.

Once a user enters Event Based Decision A and accesses the home page, Event Based Decision A becomes "active" for the remainder of the Page Flow. If the user jumps from page 1 to page 2 and then clicks the Back button, the state of page 1 is preserved. Similarly, the user can jump from page 1 to page 2 to page 3 to page 2 to page 1.

Event Based Decision A must be active before Event Based Decision B can become active.

In addition, the user must access page 7 before Event Based Decision B can become active. For example, the user can jump from page 1 to page 2 to page 3 to page 7 to page 8. The user *cannot* jump from page 1 directly to page 8.

Once a user enters Event Based Decision B and accesses either link, Event Based Decision B becomes active for the remainder of the Page Flow. As with Event Based Decision A, the user can click the Button button and the state of previous pages are preserved. For example, the following sequence is valid:

- 1 - 2 - 3 - 7 - 8 - 9 - 10 - 9 - 8 - 7 - 3 - 2 - 1 - 2 - 3 - 7 - 11 - 12 - 13

# Index