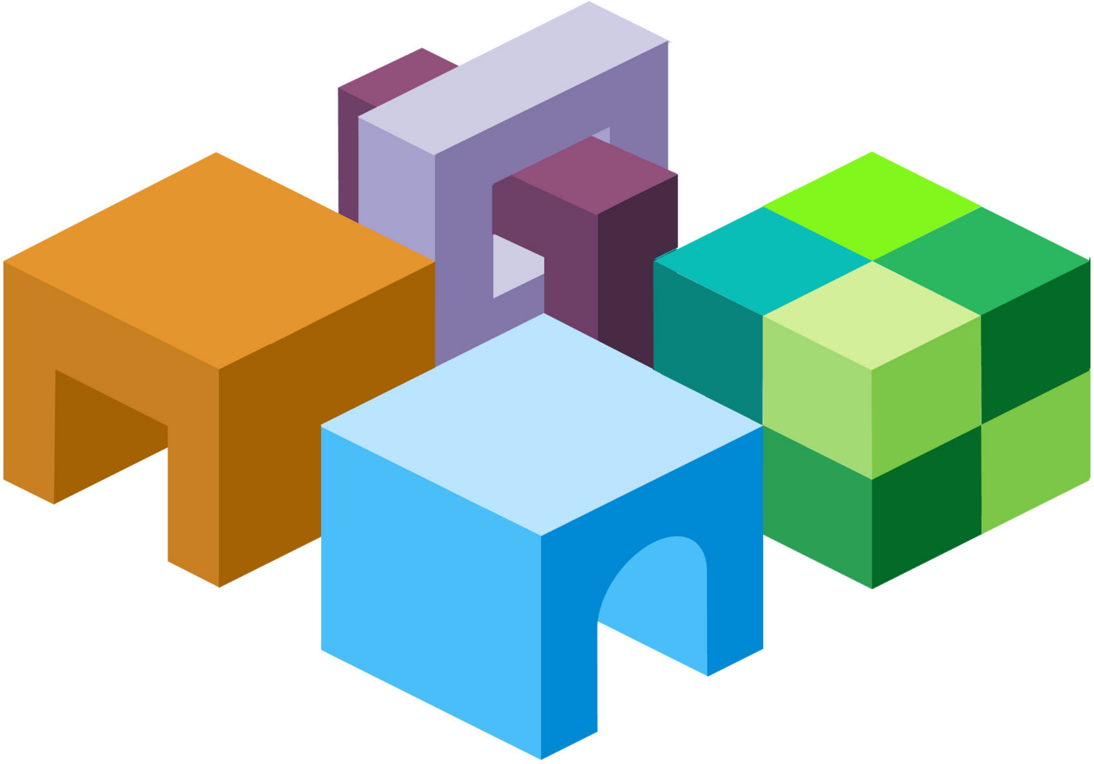


HYPERION® INTERACTIVE REPORTING

RELEASE 11.1.1

OBJECT MODEL AND DASHBOARD DEVELOPMENT SERVICES DEVELOPER'S GUIDE

VOLUME V: DASHBOARD STUDIO

ORACLE®
ENTERPRISE PERFORMANCE
MANAGEMENT SYSTEM

Interactive Reporting Object Model and Dashboard Development Services Developer's Guide, 11.1.1

Authors: EPM Information Development Team

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

This software and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third party content, products and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third party content, products or services.

Contents

Part I

Overview

In Overview:

1

About Dashboard Studio

In This Chapter

About Using Dashboard Studio

Dashboard Studio enables you to deliver the information required to monitor business metrics. In a straightforward Web format, create dynamic digital dashboards that provide real-time and scheduled access to key performance indicators, expedite operational analysis, and produce strategic management reports.

Dashboard Studio is an off-the-shelf solution that can deliver your first dashboard within a few hours of installation. Using an extensible framework, build customized dashboards and incorporate cosmetic or functional changes. Perform these tasks using Dashboard Studio:

- Specialize, extend, and override functions
- Add functions and business rules
- Match the corporate look and feel
- Add components (see the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*)

Changes produced in Dashboard Studio are available for testing immediately in the Oracle's Hyperion® Interactive Reporting Studio client application, and dashboard changes in the client application (such as creating filters, sections, or content changes) are reflected in Dashboard Studio.

Dashboards created with Dashboard Studio can be quickly deployed to suit business needs. Dashboards can be delivered over the internet, intranet, or corporate network.

Dashboard Studio is easy-to-use, and dashboards created in it are based on familiar Web site layouts, so minimal training on how to use them is required. Use Dashboard Studio to create high quality and visually enticing dashboards that present key information and metrics. Dashboards are created in eight steps with no programming required.

Compatibility Between Releases

New features and optimizations introduced with Release 9.3 in the Oracle's Hyperion® Interactive Reporting document file format, impact existing pre-9.3 Release Interactive Reporting documents saved in Interactive Reporting Studio Release 9.3 or later. These documents can lose structures when opened by an earlier release of Interactive Reporting Studio. This topic describes new structures that can be added and how documents saved in Release 9.3 or later can be made compatible with earlier releases.

Opening, using, and saving an Interactive Reporting document in Interactive Reporting Studio Release 9.3 or later without adding new structures does not cause loss of structure if opened again in an earlier release of Interactive Reporting Studio, as no new structures were added.

Examples of new structures:

New Chart Types

Scatter and bubble charts are new sections that are not recognized in earlier releases of Interactive Reporting Studio. Earlier releases ignore these sections in a Release 9.3 or later Interactive Reporting document. If that document is then saved by the earlier release, the ignored sections are not written back to the saved file and are permanently lost.

The new chart types cannot be preserved in files of earlier formats.

Image Maintenance Using the Resource Manager

The Resource Manager, an Interactive Reporting Studio Release 9.3 and later feature, restructures how images are stored; therefore, reducing the file size and memory footprint of an Interactive Reporting document and accelerating the loading speed. Image restructuring occurs when the Resource Manager is used to remove duplicate images, or if new images are added. Image restructuring also occurs if Dashboard Studio is used to merge, import, optimize, or update Interactive Reporting documents.

Note:

See “Resource Manager” in the *Hyperion Interactive Reporting Studio User's Guide.*

The general rule of backward compatibility applies to those features supported by the earlier releases of Interactive Reporting Studio. Wherever possible, new features are implemented that are available to Release 8.0 and later. However, there are exceptions:

- GIF and JPEG images are not supported by Interactive Reporting Studio Release 8.1 and earlier, use templates that include BMP images
- PNG images are not supported in releases earlier than Release 9.3 (Use the RevertImageResources.js script to convert PNG images to BMP)
- Interactive Reporting Studio Release 9.3 and later introduces the image Resource Manager to Interactive Reporting files (These Interactive Reporting documents lose their images when opened in earlier releases of Interactive Reporting Studio. Use the

RevertImageResources.js script to convert the Interactive Reporting document to the earlier format)

- Interactive Reporting Studio Release 9.3 and later introduces a new feature that enables JavaScript to create and modify dashboard controls, such as buttons and embedded charts (This feature is a prerequisite for the Webdash template, which means that Webdash can only be used to create dashboards with Interactive Reporting Studio Release 9.3 and later. However, dashboards built in Webdash can be run in earlier releases of Interactive Reporting Studio provided the RevertImageResources.js script has been run to convert the Resource Manager structures in the Interactive Reporting document to their earlier format)
- Dashboard Studio Release 9.3.1 and later introduces Filter Aliases and Filter Exclusions which change the way filters are handled internally from this release onwards (Release 9.3.1 ESM and runtime documents are now incompatible with the Move Filters feature of Dashboard Studio Optimize Utility from Release 9.3 and earlier)
- Release 9.3.1 and later introduces report embedding in dashboards (An Interactive Reporting document that contains embedded reports can be opened in an earlier release of Interactive Reporting Studio and the report section will be visible on the dashboard. However, because this feature is not supported in earlier releases, the behavior of the document may be unpredictable; for example, how the report is updated after filters are set and reset. It is recommended that documents with embedded reports are not used in earlier releases of Interactive Reporting Studio)

Files that contain new image structures can be converted to a pre-9.3 Release format by using RevertImageResources script, located in the bin folder. If the installation was not customized, the bin folder is under `C:\Hyperion\products\biplus\bin`. However, if the installation was customized, fix the path to be relative. See "RevertImageResources.js" on page 136 and "Running the RevertImageResources Script" on page 136.

The script accepts a path as a command line parameter which can be the absolute path to an Interactive Reporting document, to revert it to its pre-9.3 release format, or to a folder that contains Interactive Reporting documents, to revert them to their pre-9.3 release format.

Note:

Any backslashes (\) in the path should be duplicated. For example, `C:\\docs\\mydocument.bqy`.

The script has been used to create multiple Dashboard Development Services Update Utility kits.

- 931_update_for_85—For customers who have Release 8.5 of the Dashboard Development Services Update Utility and Interactive Reporting Studio installed
- 931_update_for_92—For customers who have Release 9.2 of the Dashboard Development Services Update Utility and Interactive Reporting Studio installed
- 931_update_for_93—For customers who have Release 9.3 or later of the Dashboard Development Services Update Utility and Interactive Reporting Studio installed (to apply Release 9.3 service packs)

- 931_update_for_other—For customers who have Release 8.x (but not Release 8.5) of Interactive Reporting Studio installed and Release 9.0.x or earlier of the Dashboard Development Services Update Utility

Also see Chapter 6, “Using the Dashboard Development Services Update Utility”, for information on the Dashboard Development Services Update Utility.

Using the RevertImageResources Script on a Folder

The example assumes that Oracle's Hyperion Reporting and Analysis is installed in `C:\Hyperion\products\biplus`, and that the folder of Interactive Reporting documents to be reverted is located in and called `Q:\files_to_revert`.

➤ To use `runRevertImageResourcesScript.bat` on a folder:

1. **In Windows, open a Command window by selecting Start, then Run, and enter** `cmd`**.**
2. **Click OK.**
3. **Enter** `cd C:\Hyperion\products\biplus\bin`**, and press Enter.**

   The path changes to the bin directory.
4. **Enter** `runRevertImageResourcesScript Q:\\files_to_revert`**, and press Enter.**

   The Interactive Reporting documents in the folder are examined, restructured, and new files are created with (images reverted) in their names. At the end of the process there are double the amount of files in the folder.

Using the RevertImageResources Script on a File

The example assumes that Reporting and Analysis is installed in `C:\Hyperion\products\biplus`, and that the Interactive Reporting document to be reverted is located in and called *`Q:\files_to_revert\my_revenue.bqy`*.

➤ To use `runRevertImageResourcesScript.bat` on a file:

1. **In Windows, open a Command window by selecting Start, then Run, and enter** `cmd`**.**
2. **Click OK.**
3. **Enter** `cd C:\Hyperion\products\biplus\bin`**, and press Enter.**

   The path changes to the bin directory.
4. **Enter** `runRevertImageResourcesScript Q:\\files_to_revert\\my_revenue.bqy`**, and press Enter.**

   The Interactive Reporting document is examined, restructured, and a new file is created called *my_revenue(images reverted).bqy*. The document can be used with earlier releases of Interactive Reporting Studio.

Note:

New features cannot be converted to earlier features. For example, if a Release 9.3 or later Interactive Reporting document contains scatter and bubble charts, these are lost when the document is opened in an earlier release of Interactive Reporting Studio.

Also see "Multiple-Release Warning" on page 231.

Dashboard Studio Templates

Dashboard Studio templates are the starting point for development of dashboards, and contain the infrastructure required by Dashboard Studio to build dashboards and for dashboards to operate at runtime. Dashboard developers use a combination of Dashboard Studio and the Interactive Reporting Studio client application to add queries, pivots, and charts to templates to produce deployable dashboards.

Dashboard Studio: A Wizard-Driven Application Builder

The Dashboard Studio application *floats* on top of Interactive Reporting Studio, and communicates with it through the client COM interface. The wizard interface and the Dashboard Studio templates automate and reduce the time required to build and maintain dashboards. Dashboard Studio consists of eight steps that enable you to point-and-click to create a dashboard. Use the wizard to quickly merge data, views, analysis, and page structure with a core framework of JavaScript functionality.

About Editable Source Master (ESM) and Runtime Dashboards

Two types of dashboards are produced using Dashboard Studio.

An editable source master (ESM) dashboard contains the code and structures required by Dashboard Studio at build time and by the dashboard at runtime. ESMs are created by default, and can be opened, modified, or enhanced by Dashboard Studio.

A runtime dashboard is a slimmer version of an ESM without build time or development structures. Runtimes are usually deployed to users and cannot be modified or enhanced by Dashboard Studio. Users are given the option to save a runtime version.

About Frame Prototypes

Frame prototypes are moulds that create consistent looking frames or skins over the analytical infrastructure of dashboards. Prototypes determine the layout and features available for frames on dashboards. See "Creating Frame Prototypes" on page 235.

A template can contain several frame prototypes with individual layouts and features. Frame prototypes available in the express template (`Express_Template.bqy`) are different to the set available in the panel templates. The Frame_Library.bqy template combines frame prototypes for the Oracle's Hyperion® Dashboard Development Services components and the panel prototypes in the one file.

In pre-8.2 Releases of Dashboard Studio, frame prototypes were called *frame templates.*

2

Using Dashboard Studio Runtime

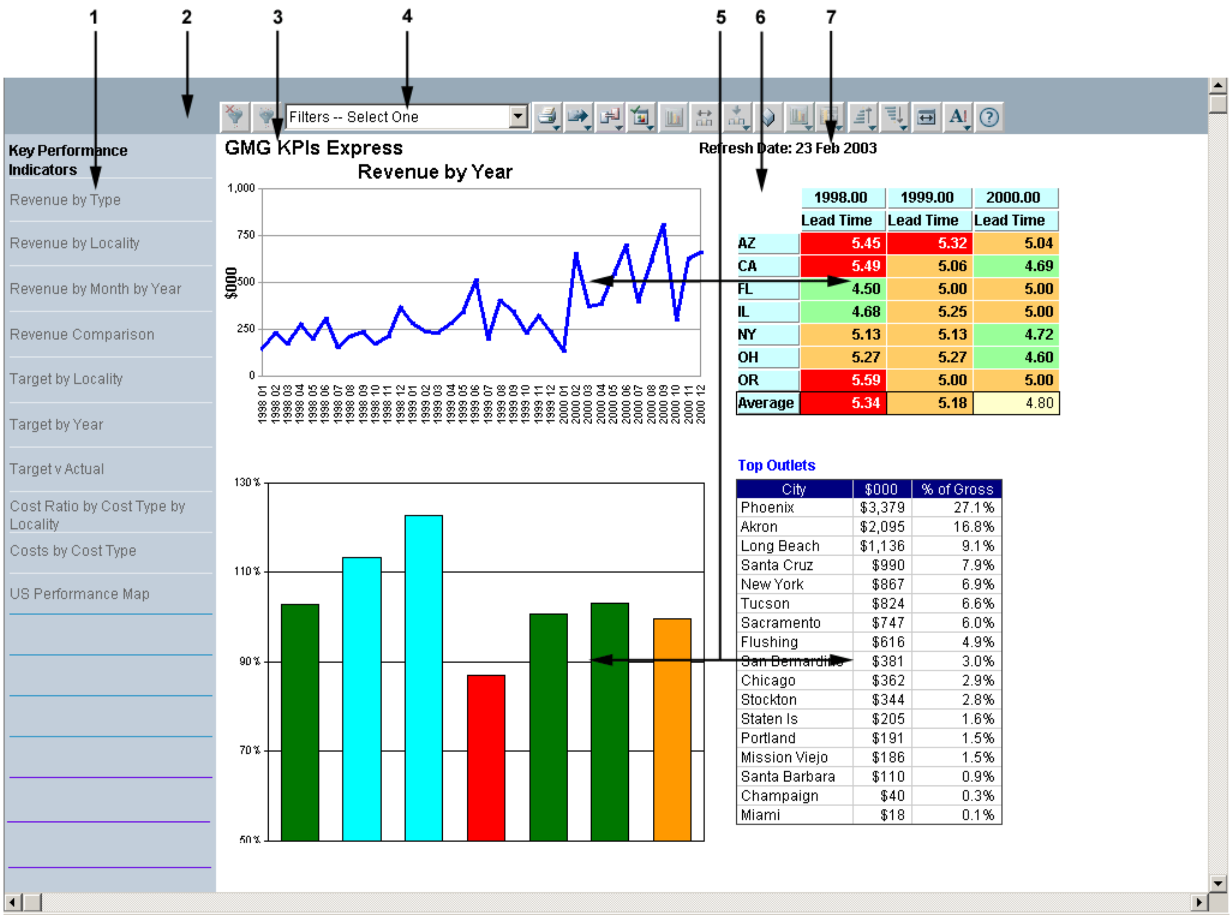
In This Chapter

Dashboard Studio Runtime Elements

Dashboard Studio runtime versions of dashboards contain features and properties of the ESM from which they are created. However, they exclude build time and development structures, and cannot be accessed or altered by Dashboard Studio. Runtime versions are most commonly deployed to users.

Figure 1 illustrates a dashboard created using Dashboard Studio. The frame uses a side-label navigation format that displays the other available frames, such as Revenue by Locality, Target by Locality, and so on. The numbered features are described in Table 1.

Figure 1 Sample Dashboard Frame



	1998.00	1999.00	2000.00
	Lead Time	Lead Time	Lead Time
AZ	5.45	5.32	5.04
CA	5.49	5.06	4.69
FL	4.50	5.00	5.00
IL	4.68	5.25	5.00
NY	5.13	5.13	4.72
OH	5.27	5.27	4.60
OR	5.59	5.00	5.00
Average	5.34	5.18	4.80

Top Outlets

City	$000	% of Gross
Phoenix	$3,379	27.1%
Akron	$2,095	16.8%
Long Beach	$1,136	9.1%
Santa Cruz	$990	7.9%
New York	$867	6.9%
Tucson	$824	6.6%
Sacramento	$747	6.0%
Flushing	$616	4.9%
San Bernardino	$381	3.0%
Chicago	$362	2.9%
Stockton	$344	2.8%
Staten Is	$205	1.6%
Portland	$191	1.5%
Mission Viejo	$186	1.5%
Santa Barbara	$110	0.9%
Champaign	$40	0.3%
Miami	$18	0.1%

Table 1 Dashboard Studio Elements

Item Label	Description
1. Navigation	Targets or hyperlinks that enable quick access to other frames in the dashboard (Side-label navigation, drop-down list, or tabbed navigation layouts are available, depending on the template used)
2. Top panel	Contains the Dashboard Studio runtime buttons for dashboards (You can place a company logo on the top left side, to customize a template)
3. Section title	Dashboard frame name
4. Filters drop-down list	Select filters to set for the dashboard
5. Views	Objects displayed in a frame (Each view is defined as View Only, Hyperlinks, or Active using standard Interactive Reporting Studio functions, see “Defining Objects as View-Only, Active, or Hyperlink” on page 58)
6. Content area	Views are manipulated and displayed in this area
7. As of date	Indicates the timeliness of the dashboard information

Self-Optimizing Runtime Dashboards

When creating a runtime document unused component sections are deleted to reduce the dashboard size and to improve performance. Therefore, an ESM contains components that can work and exist together (some components cannot share an Interactive Reporting document with others; for example, Query Limits and Quick Query Limits) without burdening the runtime with unused resources, such as code, language strings, and sections. The ESM is a developer release of the Interactive Reporting document and is less efficient, but the runtime is slim and provides the configured functionality.

Runtimes created for production should use the packing option. See "Packing JavaScript Feature" on page 190.

Runtimes created for unit testing experimental Interactive Reporting documents, those containing new JavaScript in the form of custom components or in 1100 Custom (see "1100 Custom" on page 207), should not be packed as the diagnostics that tracing enables is used.

Dashboard Navigation

The `Express_Template.bqy` and `Frame_Library.bqy` templates offer several navigation methods. Navigate through predefined dashboard views by selecting from a hyperlink, drop-down list, or tab. These views are based on frames or other standard Interactive Reporting Studio sections (for example, charts or pivots).

Templates contain a variety of frame prototypes that determine the look and feel of the dashboard.

Note:

Available frame prototypes may vary depending on the template used.

Side-Label Navigation

Navigate by selecting a hyperlink from the side-label navigation panel on the left. Click to go to the corresponding frame or Interactive Reporting Studio section.

Tip:

To use this navigation layout, select a frame prototype which supports side-label navigation.

Drop-Down Navigation

Navigate by selecting a section from a drop-down list, to go to the corresponding frame or Interactive Reporting Studio section.

Tip:

To use this navigation layout, select a frame prototype that supports drop-down navigation or use the drop-down panel template.

Tab Navigation

Navigate by selecting a section from a content tab, to go to the corresponding frame or Interactive Reporting Studio section.

Tip:

To use this navigation layout, select a frame prototype which supports tabbed navigation or use one of the tabbed panel templates.

Setting and Changing Filters

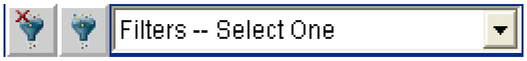
Local filters enable you to change or reduce the visible data in dashboard views. Set filters by selecting values from the Interactive Reporting Studio Filter dialog box, or select values from Quick Filters drop-down lists, check boxes, option buttons, or list boxes that are configured by the dashboard developer.

Using the Filters Drop-Down List

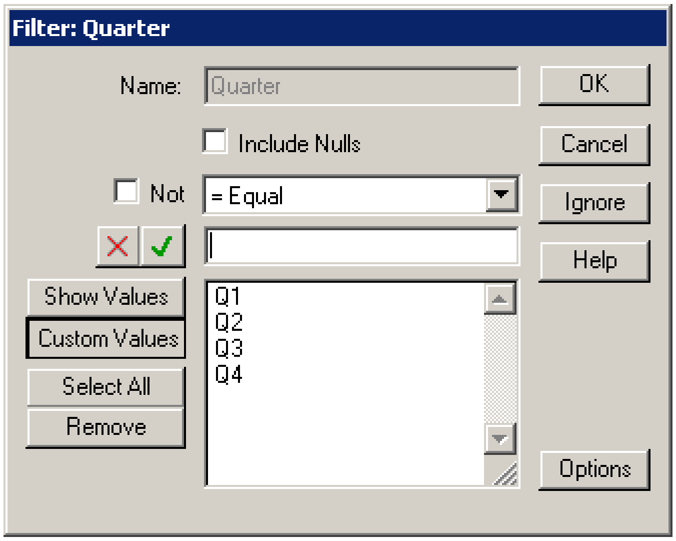
Most frame prototypes contain a drop-down list on the top panel that contains the filters for the current frame. Dashboard Studio uses regular Filter dialog boxes to set and reset filters. Operators such as Equals and Between, and functions such as Ignore work in the standard Interactive Reporting Studio way.

➤ To set or change filters:

1 **In Interactive Reporting Studio, click the Filters drop-down list, and select a filter.**



The Filter dialog box is displayed that contains available values.



Filter: Quarter

Name: Quarter

□ Include Nulls

□ Not = Equal

Q1
Q2
Q3
Q4

2 **Select the values to be set, and click OK.**

Views change according to the set filters. Filters are applied globally to sections that contain filters.

If the *Is Null* operator is set, an available value must be selected before clicking OK. The Filter dialog box enables you to select the *Is Null* operator without selecting a value, but in a dashboard, it is equivalent to clicking Cancel.

Caution!

Canceling the Filter dialog box can be an expensive operation if a lot of values are selected. An alternative and faster way to cancel the Filter dialog box is to select Not, set the <> *Not Equal* operator, select a value, and click OK.

About Quick Filters

Use Quick Filters to set filters with less clicks than the regular Filter dialog box. Dashboard Studio templates enable you to set filters using drop-down lists, check boxes, option buttons, lists, or text boxes that are configured by the dashboard developer. See "Quick Filters" on page 59 and "Configuring Quick Filters" on page 64.

About Drop-Down Lists

To filter values using drop-down lists, select a value. Views change according to values set. Only one value at a time can be selected from the drop-down list. The Ignore command clears all values set for the filter and reverts the view to the original state. See "Configuring Quick Filters Drop-Down Lists" on page 64.

About Check Boxes

To filter values using check boxes, select a value. Views change according to values set. One or more check box values can be selected. See "Configuring Option Buttons or Check Boxes" on page 65.

About Option Buttons

To filter values using option buttons, select a value. Views change according to the values set. Only one value at a time can be selected with option buttons. An option button that applies all values for a filter can be configured by the dashboard developer. See "Configuring Option Buttons or Check Boxes" on page 65.

About Text Boxes

To filter values using text boxes, select a value, and click the associated command button. Configure command buttons to activate selected filters in text boxes. See "Configuring Command Buttons" on page 66. Press Ctrl and select multiple values. See "Configuring Quick Filters Text Boxes" on page 65.

Using Lists

Quick Filters lists require selected values in one or more lists and a command button to be clicked to activate selected filters. One command button can activate all selected values or one command button can be configured for the list.

See "Configuring Command Buttons" on page 66.

➤ To filter values with lists:

1 **In a dashboard, select filter values from one or more lists.**

One or more list values can be selected. Selecting values from multiple lists and clicking the command button once is the most efficient method in terms of the work that Interactive Reporting Studio must do to apply filters. However, if the data is unfamiliar, selections filter out all rows and result in no data being displayed. For example, selecting Country = France and City = New York and clicking the command button results in no data being displayed.

Therefore, if the data is unfamiliar, select values from one list and click the command button before selecting values from another list, causing the list to update the values. For example, selecting Country = France, and clicking the command button causes the City list to show only cities in France.

Selecting Ignore from the list clears the set values.

2 **Click the command button to set Quick Filters.**

If multiple command buttons are available on the frame, click the button associated with the selected filter value. Views change according to filters set.

Set filter values are displayed in Active Filters, which is visible by default for frame prototypes with drop-down or tabbed navigation and Quick Filters. In frame prototypes with side-label navigation, Active Filters replaces the navigation bar when printing the frame. See "Printing Frames or Objects" on page 30.

If a filter is selected from Active Filters, the Filter dialog box is displayed. The dialog box enables the filter values to be modified.

Setting Filters Using Webdash-Configured Documents

Webdash-configured documents include a drop-down list that contains all the configured local filters, and a dynamic list box which displays all available filter values that are selected in the drop-down list.

➤ To set filters in a Webdash-configured document:

1 **From the drop-down list, select a filter.**

All available filter values are displayed in the list box.

2 **From the list box, select a value.**

3 **Click Set Filter.**

The dashboard data changes to reflect the selected filter.

Dashboard Studio Runtime Buttons

A standard set of buttons is available on the top panel of a dashboard created with Dashboard Studio.

Table 2 Button Descriptions

Buttons	Description
✕	Clear all filters
▼	Show filters or move to Filter Control EIS to review, clear, or set filters
🖨	Print selected items or frames Interactive Reporting Studio (It differs from Interactive Reporting Studio Print that enables the entire frame to be printed)
➡	Export sections from a dashboard into non-Interactive Reporting Studio formats, such as Excel, tab- or comma-separated, HTML, JPEG, and PDF
⇄	Activate a related document file, folder, or URL, or an e-mail message
✓	Save the current set of applied filters or apply a previous set of filters

Buttons	Description
	Switch from a chart to pivot view or a pivot to a chart view (Buttons change depending on the section in view)
	Change the size of pivot columns so that values within them are not truncated
	Specify pivot fact columns to be visible or invisible
	Switch a chart from 2-D to 3-D view or 3-D to 2-D view (Buttons change depending on the chart in view)
	Change the chart type to another type
	Show, suppress, or switch the legend from one axis to another
	Sort charts, pivots, tables, or results sections in ascending or descending order
	Switch from Wide view to Standard view and Standard view to Wide view (Buttons change depending on the view mode)
A!	Select the language in which the dashboard text (for example, floating comments and form labels) is displayed
?	● Activate help or floating comments and annotations, which detail the main dashboard navigational and functional elements ● Display or hide release information or annotations
	Present the specified Home section
©	Displays the Dashboard Studio Copyright information (Available from the Filter Control EIS frame)
<	Move to the last active frame
	Select the Diagnostics frame (Available from the Copyright frame)

Note:

Because dashboards created in Dashboard Studio are built from a customizable and extensible framework, runtime buttons and features described in the table may not be present in the dashboard you are working with. Alternatively, certain features in your dashboard may not be documented in this guide.

Managing Filters

This topic examines the Dashboard Studio runtime filter feature. Filters are configured and maintained using the Filter Control EIS frame.

The Filter Control EIS Frame

Two filter buttons on the top panel enable you to manage dashboard filters. If no filters are set, the buttons are not selectable. (Show Filters) is active if Filter Control EIS is set as the dashboard startup frame.

If the operator is set to *Is Null*, an available value must be selected before clicking OK. *Is Null* can be selected without selecting a value, but in a dashboard, it is equivalent to clicking Cancel.

To use the Filter Control EIS function, click Filter Control EIS is displayed with a list of currently applied filters.

Use Filter Control EIS to manage the filters set for a dashboard. From the frame, all local filters can be viewed, and you can set, clear all filters or individual filters. You can set additional filters for a dashboard with Available Values or Custom Values. In this respect, Filter Control EIS behaves the way of the standard Interactive Reporting Studio Filter dialog box.

To clear individual filters, select the filter, and click Ignore Selected Filters. To clear all filters, click Ignore All Filters, or click (Clear Filters) on the dashboard top panel.

To retain the filters even after closing the dashboard, select Remember Filter settings on startup. Otherwise, the dashboard opens in its original state.

Using Filter Control EIS

Filter Control EIS can be configured to use available or custom values as filters.

➤ To select filters from Available Values:

1 **In Interactive Reporting Studio, click .**

Filter Control EIS is displayed.

2 **Select the Filters drop-down list.**

A list of available values is displayed.

3 **Select a filter.**

4 **Select Available Values.**

5 **Select one or more available values.**

6 **Optional:**

- To select all available filter values, click Select All

- To set selected values as custom values, click Add to Custom Values

7 **Click Set Filter.**

The selected filters are set and the conditions are displayed in the list below Remember Filter settings on startup.

➤ To select filters from Custom Values:

1 **Select the Filters drop-down list.**

Custom values are displayed.

2 **Select a filter.**

3 **Select Custom Values.**

4 **Enter a custom condition to add to the list of custom values, and click Add.**

5 **Optional: If a date filter is selected, a calendar control is displayed.**

6 **From the calendar, select a date and time for the custom date value.**

7 **Select one or more custom values from the list.**

8 **Optional:**

- To select all available filter values, click Select All
- To remove selected values from the list, click Remove Value

9 **Click Set Filter.**

The selected filters are set and the condition is displayed in the list below Remember Filter settings on startup.

Dashboard Studio Runtime Features

This topic provides details of the Dashboard Studio runtime features.

Printing Frames or Objects

Print frames or objects within a frame.

➤ To print frames or objects:

1 **In Interactive Reporting Studio, on the dashboard top panel, click 🖨.**

A drop-down list is displayed.

If only one printable frame or object is configured, clicking 🖨 automatically launches the Print dialog box.

2 **Select an object to print.**

The Print dialog box is displayed.

When printing a frame with side-label navigation, the navigation panel is replaced with a list of currently set filters. The list is included in the printed image.

3 **Select the printer or print driver.**

For example, Adobe PDFWriter.

This feature is not available if previewing the dashboard in the Oracle Enterprise Performance Management Workspace, Fusion Edition.

4 **Click 🖨 again, to hide the drop-down list.**

Exporting Objects or Sections

Export sections from a dashboard into non-Interactive Reporting Studio formats such as Excel, tab- or comma-separated, HTML, JPEG, and PDF.

➤ To export objects or sections:

1 **In Interactive Reporting Studio, on the dashboard top panel, click ➡.**

A drop-down list is displayed.

If no objects are configured to export, ➡ is not selectable. If only one object or section is configured, clicking ➡ launches Export Section.

2 **Select an object to export.**

Export Section is displayed.

3 **Select a destination location for export.**

4 **Select a format, and click Save.**

Depending on which Interactive Reporting Studio Release is being used, formats may vary.

5 **Click ➡ again, to hide the drop-down list.**

Other rules apply if you are using EPM Workspace.

➤ To export objects or sections in EPM Workspace:

1 **Open a dashboard imported into EPM Workspace, click ➡ to export.**

EPM Workspace Export is displayed.

After exporting the section, the Web browser is launched so the exported file can be viewed and saved locally.

2 **From Section Names, select a section.**

3 **Select a File format, and click Export.**

If the server product, Release 8.1 or earlier is run, sections can only be successfully exported from EPM Workspace if values are set for the EPM Workspace export path and EPM Workspace URL path options in “Step 7: Configure Properties” on page 75.

To export to Excel or Lotus 1–2–3 formats, EPM Workspace must be configured to provide MIME types, as this setting is not the default. Configuring MIME types is outside the scope of this guide.

Dashboard Studio enables embedded sections to be exported to Microsoft Office HTML and Microsoft Office Web Archive (.MHTML) formats on EPM Workspace.

Note:

This feature is available in EPM Workspace; however, exporting to Microsoft Office Web Archive may fail due to Internet Explorer security restrictions. Therefore, export the document as Microsoft Office HTML, and save as a Web Archive when open in Internet Explorer.

Opening Related Documents

Open a file, folder, URL, or generate an e-mail that is relevant to the dashboard.

If a related document is not set in QIQ Related Document Properties, is not selectable. See “Using the QIQ Related Document Properties Frame” on page 90.

➤ To open a related document:

1 **In Interactive Reporting Studio, on the dashboard top panel, click .**

A drop-down list is displayed.

If only one related document is configured, clicking launches the related document.

2 **From the drop-down list, select a related document.**

For example, selecting http://www.hyperion.com, launches the Oracle Web site in a Web browser.

3 **Click again, to hide the drop-down list.**

Using Checkpoint Control

A checkpoint is a snapshot of settings at a point in time. Users can set checkpoints, revert the state of the dashboard to a prior point, or move to a forward point. Calls from the Checkpoint control are implemented by filters and components to provide complete restoration. See *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

Users can rename checkpoints, which are saved as *Checkpoint1, Checkpoint 2,* and so on. Checkpoints are persistent throughout the document, so these are still available after the document is saved and closed.

➤ To use the Checkpoint control to save filters that are currently set on the dashboard, or apply a set of previously saved filters:

1. **In Interactive Reporting Studio, on the dashboard top panel, click .**

   A drop-down list is displayed.

   The Starting Checkpoint command in the drop-down list applies starting filters that are configured in QIQ Filter Properties. If Remember filter settings on startup is activated from Filter Control EIS, the filters set when the dashboard was closed are applied when Starting Checkpoint is selected.

2. **Select Add Checkpoint to apply the current filters and chart properties.**

   The settings are saved as Checkpoint 1, Checkpoint 2, and so on.

3. **Optional: To return to the starting checkpoint, select Starting Checkpoint.**

   The checkpoint set at startup is displayed.

4. **Optional: To modify a checkpoint, select Edit Checkpoints.**

   An editing frame is displayed. See Editing Checkpoint Controls.

5. **Optional: To remove checkpoints, select Clear Checkpoints.**

6. **Click again, to hide the drop-down list.**

Editing Checkpoint Controls

➤ To use the editing frame to modify a Checkpoint control:

1. **Click , and select Edit Checkpoints.**

   Checkpoint Properties is displayed.

2. **From Available Checkpoints, select an item, and enter a Checkpoint Name.**

3. **Click to rename the checkpoint.**

   If another checkpoint is selected from the list, Checkpoint Name is not updated. To display the name of the selected checkpoint in Checkpoint Name, click . The entry in Checkpoint Name takes precedence over the selected Available Checkpoints.

4. **Optional: Perform an action:**

   - To delete the selected Available Checkpoints, click

- Reorder selected Available Checkpoints, by selecting a checkpoint and clicking ▲ or ▼

5. **In Checkpoint Description, enter text, and click to retain the description.**

   For example, enter `Market: Commercial Year: 2000 Quarters: 2 & 4.`

6. **To return to the dashboard frame, on the top panel, click ◀.**

7. **To view an applied checkpoint, click .**

8. **From the drop-down list, select the checkpoint.**

   The frame refreshes to display the checkpoint values.

Tip:

To refresh an edited checkpoint, from the drop-down list select another checkpoint or the starting checkpoint, and then select the edited checkpoint.

Switching Between Chart and Pivot Views

Switch between a chart and pivot view, when charts, pivots, or tables are placed on top of each other. The feature is available only if Allow Chart/Pivot Switching is set for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68. If the frame does not contain an embedded chart, pivot, or table, the feature is unavailable.

The button varies depending on whether a chart, pivot, or table is displayed. The object that is displayed when the dashboard is saved, is the object displayed when the dashboard is reopened.

In this context, a table is treated like a pivot.

Note:

If you are using a Webdash-configured document, this feature is automatically enabled when a chart and pivot, or chart and table are placed in the same panel.

Auto Sizing Column Widths

Change the size of pivot or table columns so that values are not truncated. Values displayed as # indicate that auto sizing is required.

Use the feature after values are changed so that they no longer fit properly within a column. Auto sizing is available only if Allow Auto Size of Columns is set for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68. If the frame does not contain an embedded pivot or table, or if a chart is displayed, the feature is unavailable.

Note:

If you are using a Webdash-configured document, this feature is automatically enabled when a pivot is placed on a panel.

➤ To auto size columns of a pivot or a table:

1 **In Interactive Reporting Studio, on the dashboard top panel, click .**

A drop-down list displays the dashboard tables, pivots, and columns.

2 **Select a table, pivot, or column to auto size.**

The selected items are resized so the values are displayed correctly.

3 **Click again, to hide the drop-down list.**

Tip:

To auto size all pivot or table columns, from the drop-down list, select All Columns.

Switching Pivot Facts

Control the visibility of pivot fact columns. Dashboard developers specify pivot fact columns to be visible or hidden when a dashboard is opened. End users have the option to hide or reveal additional fact columns.

Depending upon how the dashboard was configured, all fact columns available for a pivot can be displayed in a list or drop-down list.

The feature does not affect charts.

➤ To switch pivot facts:

1 **In Interactive Reporting Studio, on the dashboard top panel, click .**

- If configured, a Switch Pivot Fact list is displayed with a list of fact columns
- If a Switch Pivot Fact list is not configured, a drop-down list of fact columns is displayed

  Hidden fact columns are denoted by [ ]. Visible fact columns are denoted by [=].

2 **Select a fact column from the list.**

The column is displayed or hidden accordingly.

3 **Click again, to hide the drop-down list.**

See “Using the Pivot Fact Properties Frame” on page 94.

Note:

If a pivot is configured with the Switch Pivot Facts feature, items on the y-axis (fact) cannot be changed by Quick Slice.

Switch Pivot Facts can be used in conjunction with the Drill and Slice component that is available with Dashboard Studio. If you drill down into data and subsequently create a greater number of displayed fact columns in a pivot, Switch Pivot Facts can provide a snapshot of the most pertinent facts, by hiding the less apt information and revealing the more important facts in selected pivot fact columns. See "Using a Runtime Release of the Drill and Slice Component" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

Switching Charts Between 2-D and 3-D Views

Charts on a frame can be switched from two-dimensional to three-dimensional, or three-dimensional to two-dimensional views, by clicking ▣ (2-D) or ◈ (3-D) on the dashboard top panel. The button varies depending on whether a two-dimensional or three-dimensional chart is displayed. The feature is available only if Allow 2D/3D Switching is set for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68. If the frame does not contain an embedded chart, the feature is unavailable.

The Spotlight feature takes precedence over this feature. Therefore, you cannot switch a spotlighted chart from 2-D to 3-D. See the "Chart Spotlighter Feature" on page 40.

The switching charts between two-dimensional and three-dimensional views feature works with the Checkpoint control to store or restore the chart view on frames, section by section. See "Using Checkpoint Control" on page 32.

Note:

This feature is automatically available in Webdash-configured documents that contain charts.

Changing Chart Types

When viewing a chart, change the current chart type. The feature is available only if Allow Chart Type Switching option is set for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68. If the frame does not contain an embedded chart or if a pivot is displayed, the feature is unavailable.

The Spotlight feature takes precedence over this feature. Therefore, you cannot switch the chart type of spotlighted charts. See the "Chart Spotlighter Feature" on page 40.

Note:

This feature is automatically available in Webdash-configured documents that contain charts.

➤ To change the chart type:

1 **In Interactive Reporting Studio, on the dashboard top panel, click .**

A drop-down list is displayed.

2 **Select a chart type.**

The chart changes according to the selected chart type.

- If the chart contains one or more items on the z-axis, the pie format is not available, because that format uses only x- and y-axes (slice and fact)
- To turn a chart into a cluster format, the chart must first use one of the stacked types (for example, horizontal, vertical, or area) before it can be switched to the bar-cluster format

3 **Click again, to hide the drop-down list.**

The change chart types feature works with the Checkpoint control to store or restore chart types on frames, section by section. See "Using Checkpoint Control" on page 32.

Also see "New Chart Types" under "Compatibility Between Releases" on page 16.

Controlling Legends

When viewing a chart, turn chart legends on or off, or change the legend axis. The feature is available only if Allow Legend Switching is set for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68. If the frame does not contain an embedded chart or if a pivot is displayed, the feature is unavailable.

The Spotlight feature takes precedence over this feature. Therefore, you cannot switch the chart legend of spotlighted charts. See the "Chart Spotlighter Feature" on page 40.

Note:

This feature is automatically available in Webdash-configured documents that contain charts.

➤ To hide, show, or change the orientation of the legend:

1 **In Interactive Reporting Studio, on the dashboard top panel, click .**

A drop-down list is displayed.

2 **Select a legend control.**

The legend setting changes according to the control set.

3 **Click again, to hide the drop-down list.**

Sorting

Configure sort specifications for columns or dimensions of charts, pivots, results, or tables. The feature provides a powerful point-and-click interface with which to create and maintain sort specifications.

Sort includes a dynamic sort function, that organizes row labels or x-axes, and contains buttons. Alternatively, items can be made available for sorting.

Dashboard frames with sort specifications applied contain ↑ and ↓ on the top panel, that determine whether the sections are sorted in ascending or descending order. If default sorting is not enabled in QIQ Sort Properties, the option is unavailable.

Note:

This feature is automatically available in Webdash-configured documents.

➤ To sort in ascending or descending order:

1 **In Interactive Reporting Studio, perform an action:**

a. To sort in an ascending order, click ↑.

b. To sort in a descending order, click ↓.

A drop-down list of configured sort specifications is displayed. The columns in the drop-down list are identical regardless of the button selected. The available sort columns format varies depending on the sort option applied.

2 **Select a column to sort from the available list.**

The selected column is sorted in ascending or descending order, depending on the button that is selected. See "Using the QIQ Sort Properties Frame" on page 94.

3 **Click ↑ or ↓ again, to hide the drop-down list.**

Switching Between Wide View and Standard View Modes

↔ ☰

View matching charts and pivots in Wide View mode or Standard View mode.

Standard View mode is available on typical dashboards. In this view, various items occupy the left side, including Quick Filters, Quick Slice, or navigation items. Charts and pivots are displayed to the right in the content area of the frame.

In Wide View mode, charts and pivots occupy the entire width of the frame, covering the items on the left side. To configure this mode for charts and pivots, see "Configuring Wide View Mode for Charts and Pivots" on page 57.

The feature is available only if the frame contains configured and equivalent *wide* and *standard* objects. See “Wide View Properties ” on page 91.

The button varies depending on the mode that is displayed.

For information on the Wide View mode feature when creating frame prototypes, see “Creating Frame Prototypes” on page 235.

Controlling Languages

Dynamically change the language of the operational text in a dashboard (for example, floating comments and form labels). A variety languages are available.

➤ To change the language setting:

1 **In Interactive Reporting Studio, on the dashboard top panel, click A!.**

A drop-down list is displayed. Available languages vary depending on the template release being used.

2 **Select a language.**

The language strings are set according to the language chosen. The feature changes the strings only for floating comments and form labels. Data and data labels remain in the stored language.

Using Help

➤ To use Help in a dashboard running in Interactive Reporting Studio:

1 **On the dashboard top panel, click ? to activate help.**

A floating comment description is displayed on the top panel.

2 **Click an element on the dashboard to display the description.**

3 **Click ? again, to exit help mode.**

Information and annotations are displayed or hidden when ? ? is clicked. Button comments are displayed when the mouse is hovered over the button.

Note:

Annotations are used to describe unique elements in a template or dashboard. When in Help mode, annotations included for a frame are made visible with the comments. Annotations are available only if they are configured for a frame. See “Using the Annotation System” on page 193

Auto Z Axis Feature

Set the first filter for a frame to be added as the z-axis (depth) of charts and column labels of pivots. The feature is available only if Allow first filter to be added to Z/Top Axis is set for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68.

The number of filter values set must be lower than the specified threshold, to be visible in the embedded chart or pivot. The threshold is set in "Step 7: Configure Properties" on page 75.

If set filter values exceed the specified threshold, a default Auto Z column is displayed to signify the presence of a z- or top (column) axis. See "Using Auto Z" on page 195.

If Auto Z is enabled, do not use Quick Slice to add the first filter column on the z-axis (depth) of that frame.

Chart Spotlighter Feature

Charts and shapes (for example, circles) can be color-coded to represent trends in the data. The feature spotlights only vertical, horizontal, and pie charts. See Chapter 12, "Spotlighting Charts".

To create and maintain a set of palette definitions within Interactive Reporting documents, use the Color Palette Properties frame. See "Using the Color Palette Properties Frame" on page 98.

Chart Spotlighter prevents other features from being applied that may cause the charts to lose color. These runtime features cannot be applied to spotlighted charts:

- Switching between two-dimensional and three-dimensional charts
- Changing chart types
- Switching the legend
- Applying the Chart Shade component

3

Dashboard Studio Features

In This Chapter

Dashboard Building Process

Interactive Reporting Studio facilities are used to construct queries, charts, pivots, reports, and computed items that are used in dashboards. Dashboards built with Dashboard Studio are created as wrappers for the sections and are presented as manageable information packages.

Dashboard Studio, a wizard-driven application, connects to and works with Dashboard Studio templates. Templates become dashboards created with Dashboard Studio and are deployed at the end of step eight.

The dashboard building process using the Dashboard Studio wizard interface consists of eight steps:

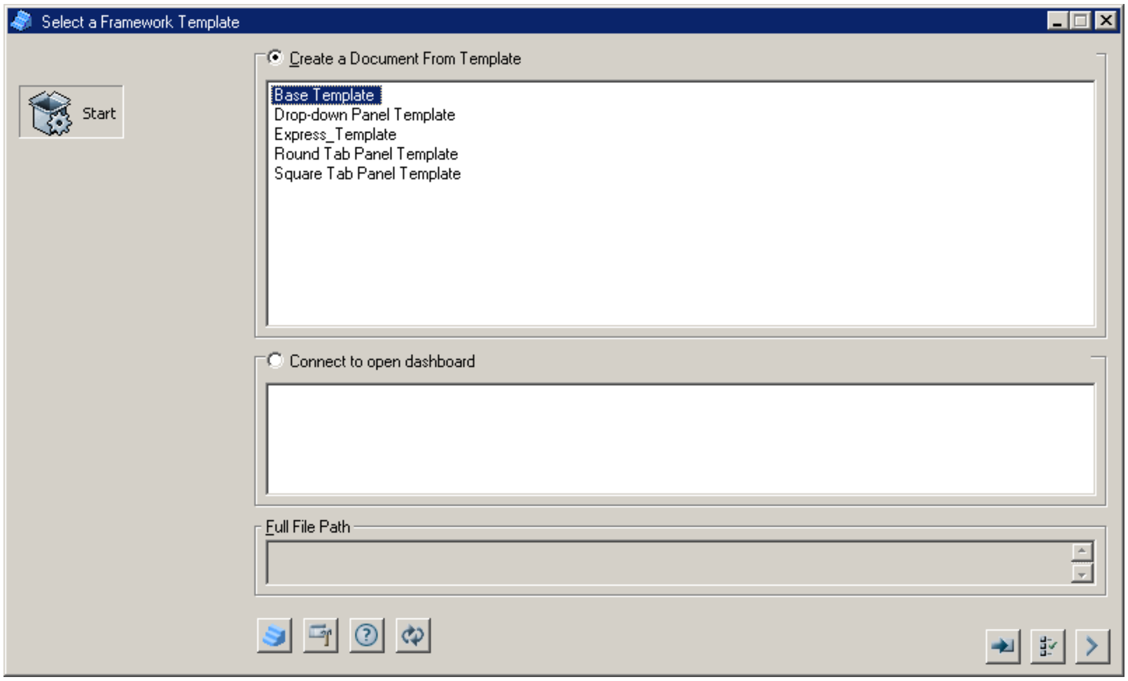
1. Queries, results, tables, charts, pivots, reports, computed items, and filters are created and named.
2. Dashboard frames are created and named.
3. Frames are populated with Interactive Reporting Studio sections.
4. Frame objects are associated with underlying Interactive Reporting Studio sections and controls are set.
5. Navigation targets are configured.
6. Filters are configured.
7. Dashboard properties are set.
8. Dashboard style elements are set.

After completing the eight-step process, save an ESM and a deployable runtime version of the dashboard.

Dashboard Studio Layout

Dashboard Studio provides icons on the left that represent the eight steps. Click an icon to proceed to the corresponding step.

Figure 2 Dashboard Studio Interface



Select a Framework Template
Start
Create a Document From Template
Base Template
Drop-down Panel Template
Express_Template
Round Tab Panel Template
Square Tab Panel Template
Connect to open dashboard
Full File Path

Dashboard Studio contains a Select a Configuration Frame drop-down list that provides access to configuration frames that are included in templates. See "Configuration Drop-Down List" on page 43.

Note:

The Select a Configuration Frame drop-down list is referred to as the Configuration drop-down list throughout the remainder of this guide.

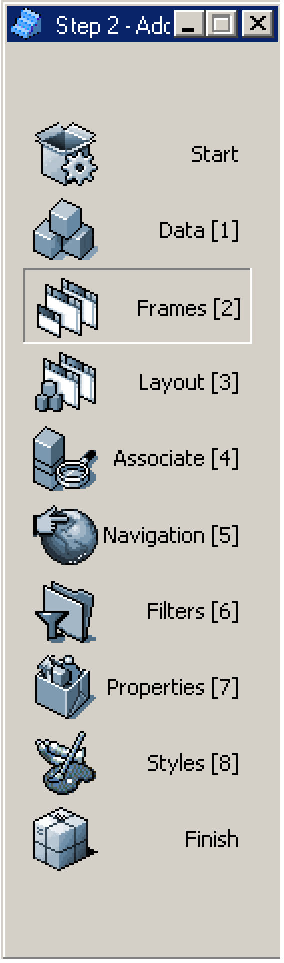
Caution!

It is recommended that only one instance of Interactive Reporting Studio is open when creating a dashboard in Dashboard Studio. If multiple instances of Interactive Reporting Studio are open, the Configuration drop-down list may activate the wrong instance.

Minimal Mode

To enable minimal mode, click the Dashboard Studio icon of the current step. Minimizing enables you to save space while creating a dashboard.

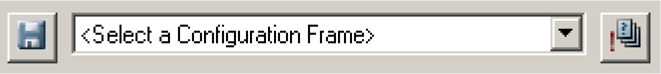
Click the icon again to maximize Dashboard Studio.



Configuration Drop-Down List

Dashboard Studio includes a drop-down list that provides quick access to the configuration frames in the template.

Selecting a section from the drop-down list puts Dashboard Studio into minimal mode and activates the selected configuration frame in Interactive Reporting Studio.



Configuration frames are part of the Dashboard Studio extensible component architecture, and provide a convenient way to configure components, some of which are core and are described in this guide, while others are optional and are described in other documentation. See the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

Configuration frames that are available vary depending on the dashboard state.

In a pre-8.3 Release template, QIQ Filter Properties is available only from the Configuration drop-down list, if the dashboard contains a populated results section.

Dashboard Studio Buttons

Available Dashboard Studio buttons.

Table 3 Dashboard Studio Buttons

Buttons	Description
✓	Apply changes or selections made in Dashboard Studio to the dashboard
⧉	Automatically create frames and provide default values for the dashboard or associate embedded objects
?	Open a context-sensitive help file (Internet Explorer needs to be installed for the help file to work)
➜	Save and close the dashboard
⚙	Display the Interactive Reporting Studio Type Library Diagnostics window (Only available from Select a Framework Template)
⇥	Import or merge sections from Interactive Reporting documents into Dashboard Studio templates
ⓘ	Make context-sensitive suggestion
>	Move to the next step
<	Move to the previous step
🎨	Open color palette
☑	Change options or settings in Dashboard Studio
⟳	Refresh or synchronize Dashboard Studio settings
≡	Display release information
💾	Save current document
💾	Save style

Dashboard Studio Steps

This topic provides an overview of Dashboard Studio steps. See “Starting Dashboard Studio” on page 48, for further details on how to configure each step.

Step 1

Work primarily with Interactive Reporting Studio to create queries, tables, charts, pivots, and reports to be used in the dashboard. Generate names for the sections that you have created.

Create the local filters to be used by the dashboard. Dashboard Studio includes QIQ Filter Properties that can create local filters, and is accessed from the Configuration drop-down list.

In a pre-8.3 Release template, QIQ Filter Properties is available only from the Configuration drop-down list, if the dashboard contains a populated results section.

Step 2

Create and name the frames of dashboards. Frames are created by duplicating a frame prototype in the template. Selected frame prototypes determine the layout and features in frames of dashboards.

Step 3

Drag Interactive Reporting Studio sections onto the frames of dashboards.

If the suggested names are accepted in Step 1 (Data), Dashboard Studio also suggests charts and pivots to be dragged on the frame.

Click ⓘ to display instructions, and identify sections to be dragged onto the frame.

Configure Quick Filters and Quick Slice objects in this step.

Step 4

Associate underlying charts, pivots, or tables with objects that you have dragged onto frames. Dashboard Studio automatically associates embedded objects or associations can be manually created.

Control settings for frames and embedded objects using these controls, switching between charts and pivots, between 2-D and 3-D views, changing chart types, controlling legends, printing of frames, and so on.

Step 5

Configure the navigation targets of the dashboard. Navigation targets enable you to move through the frames and other views in a dashboard. Frame prototypes determine the position of these targets on the frame. For example, on the side-label panels, in a drop-down list, or on tabs.

Step 6

Set filters for dashboard sections. These filters are available from the Filters drop-down list on the dashboard top panel.

If no filters are configured, move to QIQ Filter Properties using the Configuration drop-down list, to create the filters. Return to Dashboard Studio, click ⟳ to refresh settings, and configure the filters that were just created.

Optional: Filters can be created in Step 1, and configured in this step.

Step 7

Set the dashboard properties. Configure properties such as the dashboard title, chart properties, and settings for toolbars and menus.

Step 8

Set the dashboard style elements, including font and color.

After the style elements are configured, save the dashboard.

4 Using Dashboard Studio to Develop Dashboards

In This Chapter

Design Method

To maximize system benefits, before working with Dashboard Studio, determine the information to be displayed and the view relationships:

- Review the requirements for the information package
- Identify the audience

- Decide whether multiple information packages are required
- Examine the data sources and determine the queries required to return the data
- Decide what information is to be presented, and whether sections need to be built or existing sections can be used
- Build queries with aggregations that enable the return of the least number of rows
- Determine the views to be presented (pivots, charts, tables, and reports)
- Decide what filters apply to the views
- Distinguish between user filters and system filters
- Decide whether filters are to be shared over multiple data sources (that is, results and table sections)
- Determine the links used to navigate among the views
- Identify the table structures on which the views are based and the filters applied

Starting Dashboard Studio

After resolving how to arrange your dashboards, organizing the layout, and how to convey the data, begin the design process by opening Dashboard Studio.

➤ To start Dashboard Studio:

1 **Start Interactive Reporting Studio, and, on the Welcome dialog, click Cancel.**

Starting Interactive Reporting Studio before starting Dashboard Studio avoids the screen flicker caused by the intermittent display of menu bars as Dashboard Studio interacts with Interactive Reporting Studio.

2 **Open Dashboard Studio.**

Select a Framework Template is displayed.

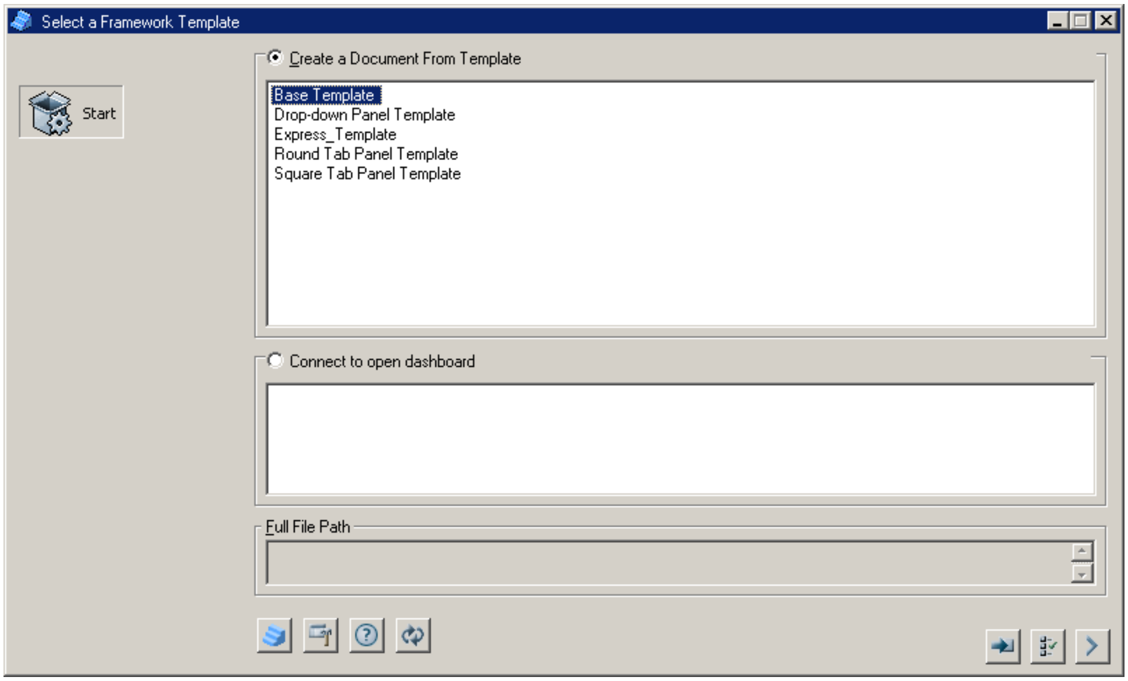
3 **Alternatively, to open Dashboard Studio, start Interactive Reporting Studio, and select Tools, then Launch Dashboard Studio.**

Select a Framework Template is displayed.

Selecting Templates and Open Editable Source Masters (ESMs)

The Select a Framework Template window is the Dashboard Studio starting point. You select a template to use to create a dashboard or connect to an open ESM to modify it in Interactive Reporting Studio.

Figure 3 Select a Framework Template



Select a Framework Template

● Create a Document From Template

Base Template
Drop-down Panel Template
Express_Template
Round Tab Panel Template
Square Tab Panel Template

○ Connect to open dashboard

Full File Path

You can only connect to an ESM, because runtime dashboards contain no Dashboard Studio build structures.

Full File Path displays the path of the selected Interactive Reporting document, which can be a template or an ESM. Roll the mouse over a listed template or dashboard to view the full path in the information box.

Tip:

Because Dashboard Studio can communicate with only one instance of Interactive Reporting Studio at a time, to display multiple ESMs, open the ESMs in the same instance of Interactive Reporting Studio.

Several controls are located on the frame, and are present on other Dashboard Studio Steps.

Table 4 Dashboard Studio Controls

Control	Description
	Provides software release information
	Assists with fixing a broken installation (Interactive Reporting Studio Type Library Diagnostics. See "Using the Type Library File to Synchronize Releases" on page 225)
?	Displays context-sensitive help
⟳	Refreshes Step settings

Control	Description
⇥	Launches Import and Merge features (see “Import Feature” on page 117)
☑	Opens the Options window (see “Options Window” on page 113)
>	Move to the next step

➤ To select a template or connect to an open ESM:

1. **From Select a Framework Template, from Create a Document From Template, select a template or from Connect to open dashboard, select an ESM.**

2. **To move to the first Dashboard Studio step, click > or double-click the selected template or ESM.**

   Step 1: Create Queries, Tables, Charts, Pivots, and Reports is displayed.

   The name of the selected Interactive Reporting document is displayed in the window title bar of Step 1: Create Queries, Tables, Charts, Pivots, and Reports. An Interactive Reporting document is a structured document that describes to Interactive Reporting Studio viewer applications how to acquire, manipulate, and present information to users. Dashboards created with Dashboard Studio are in Interactive Reporting document format.

Step 1: Create Queries, Tables, Charts, Pivots, and Reports

In this step, you work primarily with Interactive Reporting Studio to create the data: queries, tables, charts, pivots, and reports to be used in the dashboard. Names are given to each section, and filters are created.

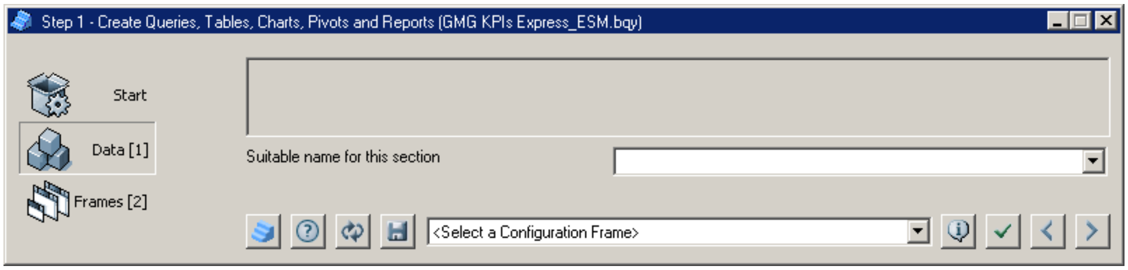
Note:

If sections are imported from an Interactive Reporting document, and no sections are added or names changed, in Dashboard Studio Step 1, click > to move directly to the next step. If sections are imported, added, or names changed, complete these procedural steps.

➤ To create and name sections, and create filters in a dashboard:

1. **From Select a Framework Template, select a template or ESM.**

   Step 1: Create Queries, Tables, Charts, Pivots, and Reports is displayed.


Step 1 - Create Queries, Tables, Charts, Pivots and Reports (GMG KPIs Express_ESM.bqy)
Start
Data [1]
Frames [2]
Suitable name for this section
<Select a Configuration Frame>

2 To display instructions, click ⓘ.

3 Minimize Dashboard Studio, and use the instructions to create Interactive Reporting Studio dashboard sections.

4 Create the filters for the dashboard:

a. From the Configuration drop-down list, select **QIQ Filter Properties**.

b. Follow the instructions provided in "Using the QIQ Filter Properties Frame" on page 83.

c. **Optional:** A table can be added to a report section to display the set filters when the report section is printed. See "Displaying Filters in Reports" on page 53.

Note:

Filter names that start with _sys (or with sys for pre-3.2 Release templates) are ignored by the filters feature of Dashboard Studio and are not altered in any way.

Filters that share a name and data type are treated as one filter, and sets of values are merged. That is, all values set for a filter are set for all results in which the filter occurs.

5 After the Interactive Reporting Studio sections are created, maximize Dashboard Studio.

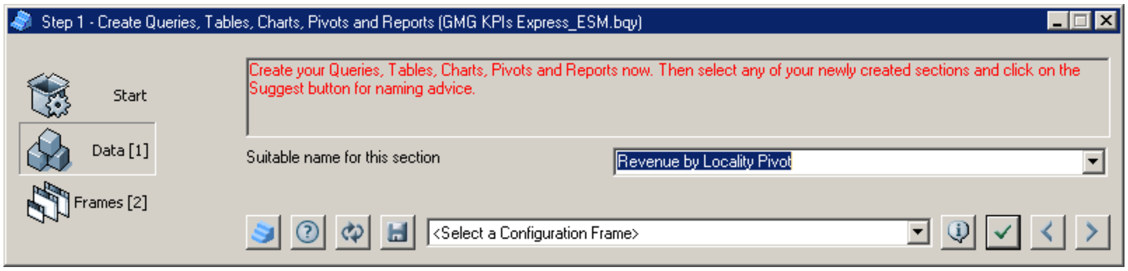
6 Use Dashboard Studio to generate names for pivots and charts.

a. In Interactive Reporting Studio, from **Sections**, select a pivot or chart.

b. In Dashboard Studio, click ⓘ to display a suggested section name.

In Suitable name for this section, a name is generated based on the first column of each part of the Data Layout.

Figure 4 Suggested Name Displayed


Step 1 - Create Queries, Tables, Charts, Pivots and Reports (GMG KPIs Express_ESM.bqy)
Start
Data [1]
Frames [2]
<Select a Configuration Frame>

Create your Queries, Tables, Charts, Pivots and Reports now. Then select any of your newly created sections and click on the Suggest button for naming advice.

Suitable name for this section: Revenue by Locality Pivot

Note:

When using a localized release of Dashboard Studio, you must click ⓘ in Step 1 for each section to ensure that localized chart or pivot equivalents are appended to the section names. If ⓘ is not clicked, clicking ▤ in Step 2 to automatically create frames may fail, because the suffix is not recognized.

c. **Optional:** In **Suitable name for this section**, enter a name.

7 **Optional: To display multiple charts in a frame, group charts by placing a prefix followed by a two-period ellipsis before the chart names (for example, *Home..Revenue Type*).**

An option described in “Step 7: Configure Properties” on page 75, removes prefixes from chart names.

8 **Optional: To display a chart or pivot in multiple frames, duplicate the chart or pivot (one for each frame).**

9 **Click ✓ to apply the selections and name the current section.**

The chart or pivot is named.

10 **Repeat steps 6–9 to name the other sections.**

11 **Click › to move to the next step.**

Step 2: Add and Rename Frames is displayed.

Defining the As of Date Column

When creating a section, you can define an As of date.

➤ To define the As of date column:

1 **In Interactive Reporting Studio:**

a. On the Request line of a query, create and name a computed item; for example, Query Date.

i. Right-click on the Request line, and select **Add Computed Item**.

Modify Item is displayed.

ii. Enter the name of the item as Query Date.

iii. In **Definition**, set the custom SQL value for the item; for example, `now()`.

iv. Click **OK** to close Modify Item.

b. Double-click the item on the Request line; for example, `now(Query Date)`, to display Item Properties.

c. Click **Options** to display the Datatype drop-down list.

d. Ensure that the column data type is Date, DateTime, or TimeStamp.

e. In **Item Properties**, click **OK** to return to the section.

2 **Click File, then Save to save the As of date.**

See “Step 7: Configure Properties” on page 75, to set the properties of the As of date column.

Displaying Filters in Reports

An active filter list is not automatically included when a report section is printed; however, when you create the section, you can configure the header and footer to include the active filter list.

➤ To display filters in reports:

1 **In Interactive Reporting Studio, from Sections, select a report.**

2 **Select Report, then Headers and Footers, and then Report Header or Report Footer.**

The header. or footer is displayed.

3 **Select Report, then Insert Table, and click in the header section.**

A table is displayed.

4 **Perform an action:**

- To display the list of active local filters, drag the `_qiqLocalFilters` column from Results to Table Dimensions
- To display the list of active variable filters, drag the `_qiqQueryLimits` column to Table Dimensions
- To display the active local and variable filters, drag the `_qiqLocalFilters` and `_qiqQueryLimits` columns to Table Dimensions

5 **To resize the table.**

a. Select the table, right-click the table header, and select Alignment.

b. From the Text Control section, select Wrap text.

The table resizes.

6 **Expand and format the table to fit the header or footer.**

Note:

The list of active variable filters is available only if the Query Limits component is merged into the current template or dashboard. See “Using a Runtime Release of the Query Limits Component” in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

Step 2: Add and Rename Frames

In this step, you create and name dashboard sections in the underlying Interactive Reporting document. These sections become the dashboard frames.

Frame Prototypes

A frame is created as a duplicate of a frame prototype available in a Dashboard Studio template. Each frame prototype provides a unique layout or functionality. See "Creating Frame Prototypes" on page 235.

Identical buttons are available across all but the Lite frame prototypes.

Note:

The Wide View feature is not recommended for Lite frames, because the navigation buttons are hidden.

Tip:

Use a combination of frame prototypes in a dashboard.

Adding and Renaming Frames

Frames can be added and renamed manually or automatically.

Note:

Some frame prototypes include improved panel descriptions; for example, 4 (2 x 2) Round |–|–|, 2 (1 x 2 Round |—|, 2 (2 x 1) Round | | |, and so on.

➤ To add or rename a frame manually:

1 **In Step 2: Add and Rename Frames, from Frame prototype to use, select a frame prototype.**

Frame prototype availability varies, depending on the Dashboard Studio template.

2 **In Total frames to create, enter the number of frames to be added to the dashboard.**

Individual frames need not be assigned for reports or other non-dashboard sections to which access is provided. Instead, use Dashboard Studio to add hyperlinks from frames to sections. See "Step 5: Configure Navigation" on page 72.

3 **Click Add.**

The added frames are listed in 1. Add new or select existing frame.

A Home frame is created by default. Rename or delete this frame if it is not applicable. The default Home frame can only be deleted if it is not the only visible section. It is advisable to add all frames and apply changes before deleting Home.

4 **Optional: To remove a frame, select the frame, and click Delete.**

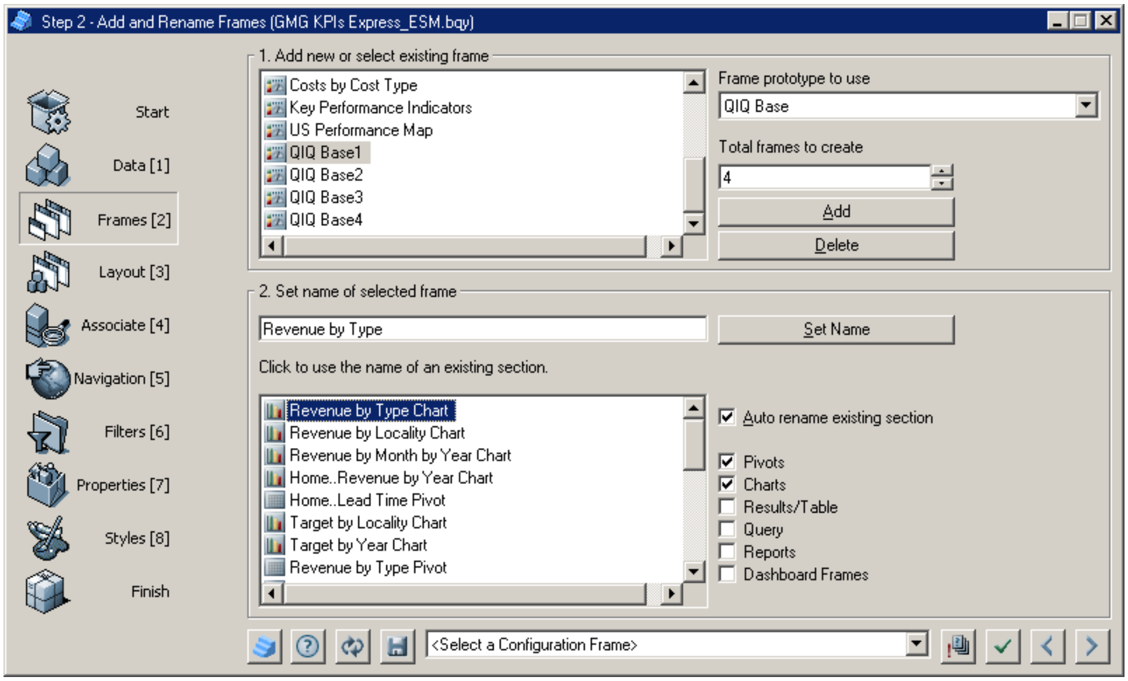
The view must contain at least one frame because the final frame cannot be deleted.

5 **After creating the frames, from 1. Add new or select existing frame, select a frame.**

The frame prototype that is selected when frames are created is used to create the frames.

6 **Name the selected frame in one of two ways:**

- In Click to use the name of an existing section, select an item
- In 2. Set name of selected frame, enter a name



Step 2 - Add and Rename Frames (GMG KPIs Express_ESM.bqy)

1. Add new or select existing frame

- Costs by Cost Type
- Key Performance Indicators
- US Performance Map
- QIQ Base1
- QIQ Base2
- QIQ Base3
- QIQ Base4

Frame prototype to use: QIQ Base

Total frames to create: 4

2. Set name of selected frame

Revenue by Type

Click to use the name of an existing section.

- Revenue by Type Chart
- Revenue by Locality Chart
- Revenue by Month by Year Chart
- Home..Revenue by Year Chart
- Home..Lead Time Pivot
- Target by Locality Chart
- Target by Year Chart
- Revenue by Type Pivot

☑ Auto rename existing section

☑ Pivots
☑ Charts
☐ Results/Table
☐ Query
☐ Reports
☐ Dashboard Frames

<Select a Configuration Frame>

7 **Click Set Name.**

Duplicate names cannot be created in Dashboard Studio. For example, if sections exist named *Revenue by Year Chart* and *Revenue by Year Pivot,* a frame is created called *Revenue by Year.*

If Auto rename existing section is selected, Dashboard Studio truncates the underlying section name automatically. For example, when Revenue by Type Chart is selected, the frame name is set to Revenue by Type.

8 **Repeat steps 5–7 to name all frames.**

9 **Click ✓ to apply the selections to the dashboard.**

The frames are created and added to the dashboard.

10 **Click > to move to the next step.**

Step 3: Drag Pivots, Charts, and Tables onto Frames is displayed.

Note:

Check boxes in the bottom-right corner enable you to filter the type of objects to be viewed. Pivots and charts are selected by default.

➤ To add or rename frames automatically:

1 **In Step 2: Add and Rename Frames, from Frame prototype to use, select a frame prototype.**

2 **Click to automatically create frames.**

Frames are automatically created based on the base name of each section. One frame is created for charts and pivots with matching base names. For example, if sections exist named *Revenue by Year Chart* and *Revenue by Year Pivot*, clicking automatically creates a frame called *Revenue by Year*.

If a section group is formed by affixing a prefix to selected section names, a group frame is created based on the prefix. Frames are not created for individual sections.

The frame prototype that is selected when frames are automatically created is used to create the frames.

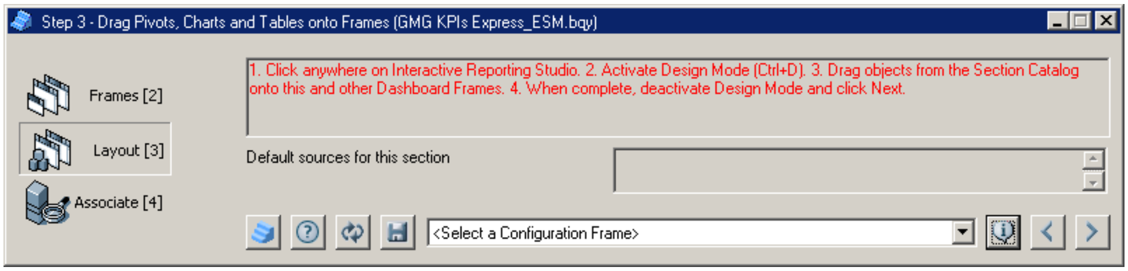
3 **Click ✓ to apply the selections to the dashboard.**

4 **Click > to move to the next step.**

Step 3: Drag Pivots, Charts, and Tables onto Frames is displayed.

Step 3: Drag Pivots, Charts, and Tables onto Frames

In this step, you drag pivots, charts, and tables onto the frames that were created in Dashboard Studio Step 2. These objects become the views that are displayed in the dashboard. Object properties are defined, views are configured, and Quick Filters and Quick Slice features are set.



Step 3 - Drag Pivots, Charts and Tables onto Frames (GMG KPIs Express_ESM.bqy)

1. Click anywhere on Interactive Reporting Studio. 2. Activate Design Mode (Ctrl+D). 3. Drag objects from the Section Catalog onto this and other Dashboard Frames. 4. When complete, deactivate Design Mode and click Next.

Default sources for this section

<Select a Configuration Frame>

Dragging Sections onto Frames

In Interactive Reporting Studio, from Sections, drag sections such as pivots, charts, and tables onto frames. After the sections are added to the frames, you can define and configure them.

➤ To drag, define, and configure sections on frames:

1 **In Interactive Reporting Studio, from Sections, select a frame.**

2 **Press Ctrl+D to enter Design mode.**

3 **From the catalog pane, drag objects to be displayed onto the blank frame.**

4 **Arrange the objects on the frame.**

5 **Press Ctrl+D to exit Design mode.**

6 **Repeat steps 1–5 to populate all frames.**

7 **Optional: Drag a chart and pivot onto the frame and position one on top of the other.**

The frame can be configured so views can switch between the chart and pivot. The last object that is dragged onto the frame is shown first in the completed dashboard. For example, if a pivot is placed first and a chart is dragged on top of it, the chart is the first object in view.

Dashboard Studio treats a table as it does a pivot. Thus, views can be switched from a chart to a table.

To switch views in a frame, the Allow Chart/Pivot Switching option must be selected for the frame in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68. Ensure that this option is not selected if chart and pivot sections are to be visible simultaneously.

8 **Set the objects to be view-only, active, or hyperlink with typical functions.**

See "Defining Objects as View-Only, Active, or Hyperlink" on page 58.

9 **In Interactive Reporting Studio, configure Quick Filters and Quick Slice objects to be available for each frame.**

See "Configuring Quick Filters" on page 64 and "Configuring Quick Slice Objects" on page 67.

10 **Optional: In Interactive Reporting Studio, configure annotations to be displayed when Help mode is active.**

See "Using the Annotation System" on page 193.

11 **Click > to move to the next step.**

Step 4: Associate Frames with Charts, Pivots, or Tables is displayed.

Note:

Report sections cannot be dragged onto frames. See "Step 5: Configure Navigation" on page 72, for instructions on how to include a report in the dashboard.

Configuring Wide View Mode for Charts and Pivots

Charts and pivots that are dragged onto frames in Standard View mode can be configured with Wide View mode.

Note:

Ensure that charts and pivots are configured *before* adding Wide View features.

➤ To configure charts and pivots with Wide View mode:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Select View, then Zoom, and then 50% to see the entire frame.**

3 **Drag objects to be displayed in Standard View mode onto the frame content area.**

4 **Duplicate the objects to be displayed in Wide View mode, and arrange the duplicate objects on the frame.**

5 **Double-click the Wide View mode object.**

Properties is displayed.

6 **Set the name of each duplicate object to match the name of the original object, and add an underscore (_) at the end of the name.**

For example, if the Standard View mode chart is called *Chart1*, the duplicated Wide View mode chart must be called *Chart1_*.

7 **Press Ctrl+D to exit Design mode.**

Tip:

An easy way to configure object names is to double-click the Standard View object, press Ctrl+C to copy the object name, double-click the corresponding Wide View object, press Ctrl+V to paste the object name, and add a trailing underscore.

Caution!

It is recommended that each Wide View object has a corresponding version in Standard View, as objects with an underscore after the name are not displayed in the Associate step of Dashboard Studio. Corresponding Wide View and Standard View objects are treated as identical objects, so features applied in the Associate and Properties steps of Dashboard Studio apply to both objects.

Also see "Switching Between Wide View and Standard View Modes" on page 38.

Defining Objects as View-Only, Active, or Hyperlink

Objects in a frame can be defined as view-only, active, or hyperlink objects.

➤ To set object properties:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Double-click the object on the frame.**

3 **Select the property to set for the embedded object, and click OK.**

4 **Press Ctrl+D to exit Design mode.**

View-Only Objects

In the completed dashboard, a view-only object can be printed, exported, toggled between two-dimensional and three-dimensional, and switched from a pivot to a chart view. You cannot drill up or down, focus and hide objects, or move pivot labels.

Active Objects

In the completed dashboard, an active object can be printed, exported, and toggled between two-dimensional and three-dimensional, and switched from a pivot to a chart view. Clicking an active object enables these actions:

- Drill down and drill up
- Focus and hide
- Move row labels to the top (for pivots)

In earlier releases of Interactive Reporting Studio, an active object could be made invisible; therefore, objects were view-only if they were switchable.

Configuring Active Charts

The procedure enables you to set an active chart to be view-only and to accommodate the entire content area of a frame.

➤ To make active charts occupy the full content area:

1. **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**
2. **Double-click the chart.**

   Properties is displayed.
3. **Select View-only and Auto-Size, and click OK.**
4. **Double-click the chart, select Active, and click OK.**
5. **Press Ctrl+D to exit Design mode.**

Hyperlink Objects

In the completed dashboard, a hyperlink object can be printed, exported, toggled between two-dimensional and three-dimensional, and switched from a pivot to a chart view. Clicking the object takes you to another Interactive Reporting Studio section. By default, the move is to the *underlying source object* or to another section chosen by the dashboard developer. The default JavaScript associated with the object must be changed.

Quick Filters

Some frame prototypes include sample Quick Filters drop-down lists, check boxes, option buttons, lists, and text boxes that can be configured to control filters.

Sample Quick Filters contain a line of JavaScript that calls the Dashboard Studio control handler for immediate setting of the filter. To delay setting filter values in list boxes, check boxes, or text boxes so that multiple selections can be applied together, remove the line of code from these controls and ensure that a quick filter button is available. Copy and paste as many objects as required, configure each one, and delete unwanted objects.

Configuring Objects for Filters

➤ To configure an object and enable it to set filters, specify the name of the filter to set and the value to apply to each object.

Table 5 Format for Object Names

Format	Object Name
`<Filter_Name>^qiqcbx`	check box
`<Filter_Name>^qiqrbt`	option button
`<Filter_Name>^qiqtxb`	text box
`<Filter_Name>^qiqdrp<suffix>^showname^a^showignore`	drop-down list
`<Filter_Name>^qiqlbx<suffix>^showname^a^showignore`	list

For example, to configure a check box so that it points to the *Quarter* filter, precede the object name with `Quarter^qiqcbx`.

For unique names, Interactive Reporting Studio adds numbers to the check box, drop-down list, option button, list, and text box names when these objects are copied and pasted. The numbers are ignored by Dashboard Studio.

Quick Filters objects can be copied to any frame, even those without sample Quick Filters objects.

Defining Optional Settings

List Boxes and drop-down lists can have optional settings added to their names to:

- Show or suppress the filter name
- Sort content in ascending or descending order
- Show or suppress the Ignore keyword

Table 6 Optional Settings for List Boxes and Drop-down Lists

Optional Setting	Object Name
`<Filter_Name>^qiqlbx<suffix>^<nameSetting>^<sortSetting>^<ignoreSetting>`	list box
`<Filter_Name>^qiqdrp<suffix>^<nameSetting>^<sortSetting>^<ignoreSetting>`	drop-down list

Table 7 Valid Values for Optional Settings

Setting	Values
`<nameSetting>`	`showname` / `hidename` (Default=`showname`)
`<sortSetting>`	`a` /`d` (Default=`a`)

Setting	Values
`<ignoreSetting>`	`showignore` /`hideignore` (Default=`showignore`)

Note:

The order of the settings is not important. Any setting can be omitted. If a setting is omitted, the default value is used.

Quick filters are named as follows:

```
<Filter_Name>^qiqlbx<suffix>^hidename^d^hideignore
```

```
<Filter_Name>^qiqdrp<suffix>^hidename^d^hideignore
```

For example, all the following would be valid and all would be identical:

```
City^qiqdrp^hidename^d^hideignore

City^qiqdrp^hidename^hideignore^d

City^qiqdrp^d^hideignore^hidename

City^qiqdrp^d^hidename^hideignore

City^qiqdrp^hideignore^hidename^d

City^qiqdrp^hideignore^d^hidename
```

The following two settings are valid and have the same effect.

```
City^qiqdrp^showname^a^hideignore

City^qiqdrp^hideignore
```

Check Boxes

Check boxes add values to the list of selected filter values. Values are removed from the list when check boxes are cleared. The set filter value corresponds to the value in the object title. You can use check boxes to set one or multiple filter values simultaneously.

If the line of JavaScript that calls the Quick Filter handler is present in the control, then each click of a check box sets or resets the selected value. If the line of JavaScript is commented out or removed from the control, then the values selected are not set or reset until the Quick Filters command button is pressed, at which time all check box, text box and list box selections are processed at the same time.

Option Buttons

Option buttons remove all selected values and add a value to the list of selected filters. The set filter value corresponds to the value in the object title. You can use option buttons to set one filter at a time.

When configuring an option button, the sample object must contain *all* as the ending suffix. The suffix sets the object to select all values for the specified filter, and the title is ignored. If an option button is set to a value, remove *all* or add other characters, and specify a value in the object title.

Drop-Down Lists

Drop-down lists remove all selected values and add values to the list of selected filters. Like check boxes and option buttons, drop-down lists must be associated with filters. You do not need to specify values, because drop-down lists load the values automatically when filters are set or when frames are activated. You can use drop-down lists to set one filter value at a time.

➤ To load values in a drop-down list in ascending or descending order, add `^a` or `^d` to the control name.

Drop-down lists can include the following additional entries.

- Filter name
- Ignore filter command

➤ To remove a filter name from the top of the drop-down list, add `^hidename` to the name of the control.

To include the filter name again, change `^hidename` to `^showname` in the name of the control, or remove `^hidename` from the control name.

➤ To remove the Ignore keyword from the top of the drop-down list, add `^hideignore` to the name of the control.

To include the filter name again, change `^hideignore` to `^showignore` in the name of the control, or remove `^hideignore` from the control name.

Text Boxes

Text boxes enable users to enter and set filters, and display information.

Lists

Lists add one or more values to the list of selected filter values. Like drop-down lists, lists load available values automatically when filters are set or when frames are activated.

➤ To set values in a list to be loaded in ascending or descending order, add `^a` or `^d` to the name of the control.

Lists can include the following additional entries.

- Filter name
- Ignore filter command

➤ To remove a filter name from the list, add `^hidename` to the name of the control.

To include the filter name again, change `^hidename` to `^showname` in the name of the control, or remove `^hidename` from the control name.

➤ To remove the Ignore keyword from the top of the list, add `^hideignore` to the name of the control.

To include the filter name again, change `^hideignore` to `^showignore` in the name of the control, or remove `^hideignore` from the control name.

Tip:

Press Ctrl and select multiple filter values simultaneously.

Tip:

For efficiency, defer list box selections to when the Quick Filter command button is pressed.

If the line of JavaScript that calls the Quick Filter handler is present in the control, then each click in the list box sets or resets the selected value. If the line of JavaScript is commented out or removed from the control, then the values selected are not set or reset until the Quick Filters command button is pressed, at which time all check box, text box and list box selections are processed at the same time.

Setting Filter Command Buttons for Lists, Check Boxes, and Text Boxes

Quick Filter lists, check boxes, and text boxes can be associated with a Set Filter command button. Use one command button to set filters for one or more list, check box, or text box, or configure a command button for each individual list, check box, or text box.

If the name of the command button begins with `qiqall` (for example, `qiqall^qiqlbxcbt`), the command button sets filter values selected from all lists, check boxes, and text boxes in the frame.

➤ To configure a command button to set filters for only one list, check box, or text box, replace the object name (`qiqall`) with the associated filter name.

For example, a command button named `Region^qiqtxbcbt` sets values from a text box called `Region^qiqtxb`.

After you select values from multiple lists, check boxes, or text boxes, do one of the following:

- Click the command button once to set all selected filters.
- Click multiple times to set values from one list, check box, or text box at a time.

Although it may be faster to set filters using the first method, if mutually exclusive conditions are selected, the first method can result in no data displayed.

Note:

You can set one or more filter values at a time. If you are not familiar with the data, it may be best to select values from one list, check box, or text box and click the command button before selecting values from another list, check box, or text box.

Configuring Quick Filters

Quick Filters properties can be configured to be displayed, hidden, or sorted in ascending or descending order.

Configuring Quick Filters Drop-Down Lists

➤ To configure a Quick Filters drop-down list:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Select a drop-down list, right-click, and select Properties.**

3 **In Name, replace *`Some_Filter`* with the filter name.**

For example, to associate a filter called *Quarter* with the object, replace *`Some_Filter`*`^qiqdrp1` with `Quarter^qiqdrp1`. The filter name must match the name of a filter created in a table or the results. In this respect, Interactive Reporting Studio is case-sensitive.

If filter names contain spaces, replace the spaces with underscores in the object name. For example, *`Year Name`*`^qiqdrp` is not a valid object, so it becomes *`Year_Name`*`^qiqdrp`.

4 **Click OK and press Ctrl+D to exit Design mode.**

Configuring Hide Properties for Column Names

➤ To hide the column name from the first line of Quick Filters in a list or drop-down list:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Select a list or a drop-down list, right-click, and select Properties.**

3 **In Name, replace `^showname` with `^hidename`.**

For example, `Quarter^qiqdrp1^`*`showname`* is replaced with `Quarter^qiqdrp1^`*`hidename`*.

4 **Click OK and press Ctrl+D to exit Design mode.**

Configuring Hide Properties for Ignore

➤ To hide the Ignore keyword from the first/second line of Quick Filters in a list or drop-down list:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Select a list or a drop-down list, right-click, and select Properties.**

3 **In Name, replace `^showignore` with `^hideignore`.**

For example, `Quarter^qiqdrp1^`*`showignore`* is replaced with `Quarter^qiqdrp1^`*`hideignore`*.

4 **Click OK, and press Ctrl+D to exit Design mode.**

Configuring Quick Filters to Load in Reverse Order

Dashboard Studio automatically loads Quick Filters in ascending order.

➤ To configure a list or drop-down list to load values in descending order, change the fourth part of the object name from `^a` to `^d`.

For example, change `Quarter^qiqdrp1^showname^a` to `Quarter^qiqdrp1^showname^d`.

Configuring Option Buttons or Check Boxes

➤ To configure Quick Filters option buttons or check boxes:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Select an option button or check box, right-click, and select Properties.**

3 **In Title, enter the associated filter value.**

For example, filter Q1 must match a value in the column or filter to which it refers. In this respect, Interactive Reporting Studio is case-sensitive.

4 **Click OK.**

5 **Copy the configured object, and paste as many instances as required.**

Configured objects can be copied to multiple frames. Configure each object by specifying the filter name and associated value.

6 **Press Ctrl+D to exit Design mode.**

Note:

You can remove the JavaScript in check box objects if you plan to defer setting the values associated with the check boxes to when the Quick Filters command button is pressed. For greater efficiency, the Quick Filters command button processes check boxes, list boxes and text boxes at one time.

Configuring Quick Filters Text Boxes

➤ To configure a Quick Filters text box:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Select the text box, right-click, and select Properties.**

3 **Optional:** If no sample text box is visible, from the catalog pane, expand **Controls** and drag a text box object into the frame.

4 In **Name**, replace `Some_Filter` with the name of the filter to be set.

For example, to associate the filter called *Quarter* with the text box, replace `Some_Filter^qiqtxb1` with `Quarter^qiqtxb1`.

The specified filter name must match the name of a filter created in a table or results. In this respect, Interactive Reporting Studio is case-sensitive.

If a filter name contains a space, replace the space with an underscore (_) when setting the text box name. For example, `Year_Name^qiqtxb`.

5 Click **OK**, and press **Ctrl+D** to exit Design mode.

Configuring Command Buttons

Command buttons can be used to defer setting filters from when a control is pressed to when the command button is pressed. A command button can set all values for check boxes, list boxes and text boxes at one time, or it can be set up to control individual list boxes and text boxes.

The advantage to deferring the setting of filters is that all selections will be done at the same time, which is most efficient. In addition, when using a pure thin-client, there is one round-trip to the server versus a trip with each click. The disadvantage is that users can make mutually exclusive selections that result in no data displayed. For example, choosing Calendar Quarter 4 and January would result in no data shown.

A command button called `qiqall^qiqlbxcbt` activates and sets all filters based on selections made in the Quick Filters controls in the active section. A command button with a prefix other than `qiqall` (for example `City^qiqlbxcbt`) activates and sets a list box called `City^qiqlbx` or a text box called `City^qiqtxb`.

➤ To configure Quick Filters so one command button sets multiple filters:

1 In Interactive Reporting Studio, press **Ctrl+D** to enter Design mode.

2 Use the command button called `qiqall^qiqlbxcbt` to set multiple Quick Filter controls.

3 Select the check box, list box, or text box to be processed by the command button; then, right-click and select **Scripts**.

4 Delete the line of script under each check box, list box, and text box for which the command button will set values.

This prevents setting the filter when the control is clicked.

5 Repeat to steps 3 and 4 for each additional Quick Filter control you want to set.

➤ To configure Quick Filter command buttons associated with each list or text box:

1 In Interactive Reporting Studio, press **Ctrl+D** to enter Design mode.

2 Double-click a command button to display **Properties.**

3 In **Name**, replace `qiqall` with the name of the associated list or text box.

For example, to set filters for Region values, replace `qiqall` with Region. In this respect Interactive Reporting Studio is case-sensitive.

4 **Enter a title to be visible to end users.**

5 **Copy and paste as many instances of the command button as required.**

6 **Select the list box or check box associated with the filter, right click and select scripts, and remove the line of code under the object.**

7 **Repeat steps 2–6 to configure each copied button.**

You do not need to configure the sample button if a command button is not assigned to each list or text box.

8 **Position the configured filter handlers in the frame.**

Filter handlers are activated when selections are applied in the documents.

9 **Click OK, and press Ctrl+D to exit Design mode.**

Quick Slice Objects

Dynamically change the content of charts (x-, y-, and z-axes) and pivots (row labels, column labels, and facts) with option buttons and check boxes configured by the dashboard developer.

Quick Slice includes three option buttons that determine the axis to be manipulated. For example, selecting the x-axis option enables you to manipulate the x-axis of charts and the row labels of pivots.

Quick Slice also includes a sample check box that enables you to select items or columns to add to charts or pivots. For example, selecting Quarter enables you to add *Quarter* to the x-axis of charts and the row labels of pivots. The object names of Quick Slice check boxes must use the structure:

```
<column>^qiqsl^<suffix>
```

where <*column*> is the column name as it occurs in the results and <*suffix*> is a number generated by Interactive Reporting Studio when controls are duplicated. The suffix is ignored by the template framework code.

Configuring Quick Slice Objects

Configure Quick Slice properties by completing these procedures.

➤ To configure Quick Slice objects:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Determine the axis that is to be manipulated; x, y, or z.**

3 **Delete unnecessary option buttons.**

Do not configure the option button controls as they are named, *x^qiqsl, y^qiqsl, or z^qiqsl.* Renaming those controls is effectively deleting them.

4. **Double-click the sample Quick Slice check box.**

   Properties is displayed.

5. **In Name, replace *`Some_Column`* with the column name.**

   For example, to associate the column called *Quarter* with the object, replace *`Some_Column`*`^qiqsl^2` with `Quarter^qiqsl^2`. The suffix is a unique number that can be ignored. The column name must match the name of a column in a table or the results. In this respect, Interactive Reporting Studio is case-sensitive.

   If column names contain spaces, replace the spaces with underscores in the object name; for example, `Year Name^qiqsl`, becomes `Year_Name^qiqsl`.

6. **In Title, replace *a column* with a name.**

   Typically, the column name is entered.

7. **Click OK.**

8. **Copy and paste as many instances of the check box as required.**

9. **Repeat steps 4–7 to configure each check box.**

   The Quick Slice objects are activated when the frames are created. See "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68.

10. **Press Ctrl+D to exit Design mode.**

Note:

Quick Slice works with the Checkpoint control to store or restore columns added to frames, section by section. See "Using Checkpoint Control" on page 32.

Step 4: Associate Frames with Charts, Pivots, or Tables

In this step, underlying charts, pivots, and tables are associated with objects that were dragged onto the frames. You can control the settings for each frame and its embedded objects.

Note:

Underlying sections must be associated because Interactive Reporting Studio does not provide association information through its object model. Interactive Reporting Studio provides clues that Dashboard Studio can use to suggest associations, but the clues may not be accurate. Suggestions made by Dashboard Studio can be overridden.

Making Associations

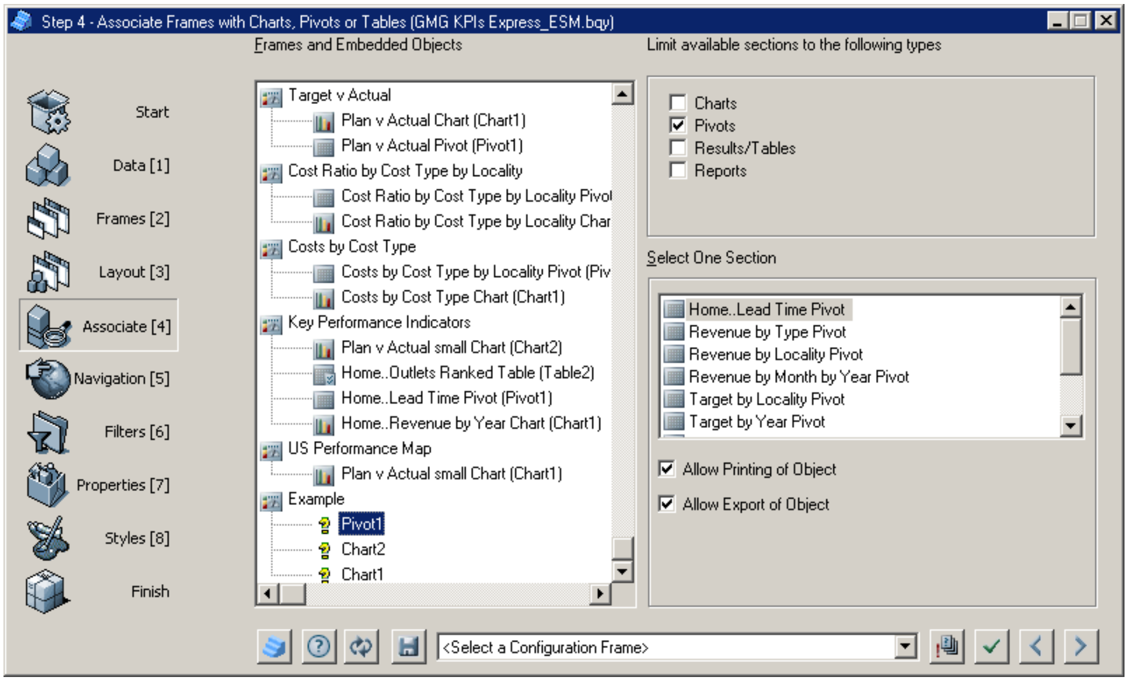
The ? indicates that the object is not yet associated with an underlying pivot, chart, or table.

➤ To make associations:

1 **In Step 4: Associate Frames with Charts, Pivots, or Tables, click to associate the embedded objects.**

Dashboard Studio associates the objects embedded on the created frames.

Figure 5 Unassociated Embedded Objects


Step 4 - Associate Frames with Charts, Pivots or Tables (GMG KPIs Express_ESM.bqy)
Start
Data [1]
Frames [2]
Layout [3]
Associate [4]
Navigation [5]
Filters [6]
Properties [7]
Styles [8]
Finish
<Select a Configuration Frame>

Frames and Embedded Objects

- Target v Actual
  - Plan v Actual Chart (Chart1)
  - Plan v Actual Pivot (Pivot1)
- Cost Ratio by Cost Type by Locality
  - Cost Ratio by Cost Type by Locality Pivot
  - Cost Ratio by Cost Type by Locality Char
- Costs by Cost Type
  - Costs by Cost Type by Locality Pivot (Piv
  - Costs by Cost Type Chart (Chart1)
- Key Performance Indicators
  - Plan v Actual small Chart (Chart2)
  - Home..Outlets Ranked Table (Table2)
  - Home..Lead Time Pivot (Pivot1)
  - Home..Revenue by Year Chart (Chart1)
- US Performance Map
  - Plan v Actual small Chart (Chart1)
- Example
  - Pivot1
  - Chart2
  - Chart1

Limit available sections to the following types

☐ Charts
☑ Pivots
☐ Results/Tables
☐ Reports

Select One Section

- Home..Lead Time Pivot
- Revenue by Type Pivot
- Revenue by Locality Pivot
- Revenue by Month by Year Pivot
- Target by Locality Pivot
- Target by Year Pivot

☑ Allow Printing of Object

☑ Allow Export of Object

2 **Click ✓ to apply the selections to the document, and agree with the associations suggested by Dashboard Studio.**

Dashboard Studio can associate most embedded objects. The automatic method of creating frames may not correctly associate an embedded object for which the code is changed or the name is changed after the object is dragged onto the frame. Interactive Reporting Studio does not update the JavaScript inside an object so Dashboard Studio cannot find a renamed object.

3 **If Dashboard Studio cannot correctly associate an object, associate the object manually:**

a. From **Frames and Embedded Objects**, select the object to be associated.

b. From **Select One Section**, select the section with which to associate the object.

4 **Optional: If the sections were manually associated, to revert to default Dashboard Studio associations, click to reassociate the sections with the frames.**

Manual associations are lost.

Using Controls for Frames

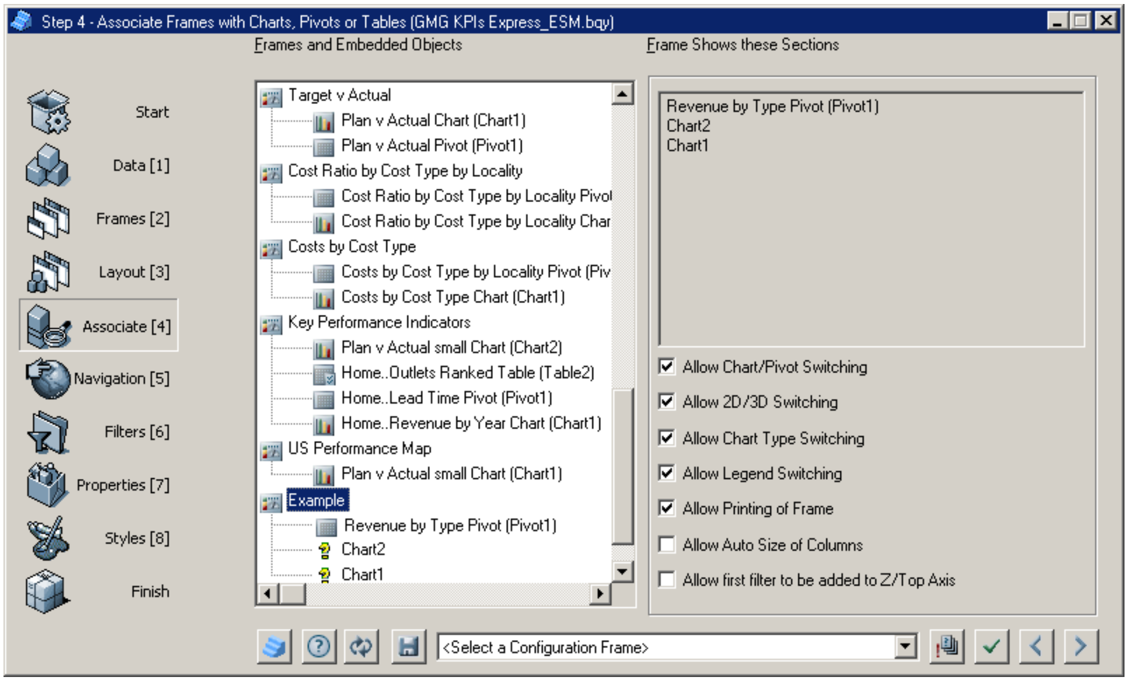
Frame controls are selected by default, with the exception of the option to enable columns to be automatically sized. The controls are unavailable if the frame does not contain an embedded chart, pivot, or table. Only two controls are available on frames created from Lite frame prototypes, these are: Allow Printing of Frames and Allow Auto Size of Columns.

- **Allow Chart/Pivot Switching**—Enables toggling from chart to pivot view or pivot to chart view within a frame (To view a chart and pivot simultaneously in a frame, clear Visible properties, this overrides the Design mode, Visible properties settings)
- **Allow 2D/3D Switching**—Enables toggling a chart view from two-dimensional view to three-dimensional view and three-dimensional view to two-dimensional view within a frame
- **Allow Chart Type Switching**—Enables changing the current chart type
- **Allow Legend Switching**—Enables turning chart legends on and off and changing the legend axis
- **Allow Printing of Frame**—Enables printing of frames
- **Allow Auto Size of Columns**—Automatically resize pivot or table columns (For example, when a drill operation changes data or when number formats are explicitly set)
- **Allow first filter to be added to Z/Top Axis**—Activates the Auto Z feature (Auto Z adds the first filter specified in the Dashboard Studio Filters step as a z-axis (depth) item to charts or a column label to pivots. The feature enables the comparison of measures. See "Using Auto Z" on page 195)

➤ To use the controls for frames:

1 **In Step 4: Associate Frames with Charts, Pivots, or Tables from Frames and Embedded Objects, select a frame.**

Frame Shows these Sections lists the objects associated with the selected frame. If the objects are not named, they are identified as *Pivot1*, *Chart1*, *Table1*, and so on.



Step 4 - Associate Frames with Charts, Pivots or Tables (GMG KPIs Express_ESM.bqy)

Frames and Embedded Objects

- Target v Actual
  - Plan v Actual Chart (Chart1)
  - Plan v Actual Pivot (Pivot1)
- Cost Ratio by Cost Type by Locality
  - Cost Ratio by Cost Type by Locality Pivot
  - Cost Ratio by Cost Type by Locality Char
- Costs by Cost Type
  - Costs by Cost Type by Locality Pivot (Piv
  - Costs by Cost Type Chart (Chart1)
- Key Performance Indicators
  - Plan v Actual small Chart (Chart2)
  - Home..Outlets Ranked Table (Table2)
  - Home..Lead Time Pivot (Pivot1)
  - Home..Revenue by Year Chart (Chart1)
- US Performance Map
  - Plan v Actual small Chart (Chart1)
- Example
  - Revenue by Type Pivot (Pivot1)
  - Chart2
  - Chart1

Frame Shows these Sections

Revenue by Type Pivot (Pivot1)
Chart2
Chart1

☑ Allow Chart/Pivot Switching
☑ Allow 2D/3D Switching
☑ Allow Chart Type Switching
☑ Allow Legend Switching
☑ Allow Printing of Frame
☐ Allow Auto Size of Columns
☐ Allow first filter to be added to Z/Top Axis

<Select a Configuration Frame>

2 Select or clear individual frame control options.

3 Click to associate the sections with the frames.

4 After the controls are configured, click ✓ to apply the selections to the document.

5 Click > to move to the next step.

Step 5: Configure Navigation is displayed.

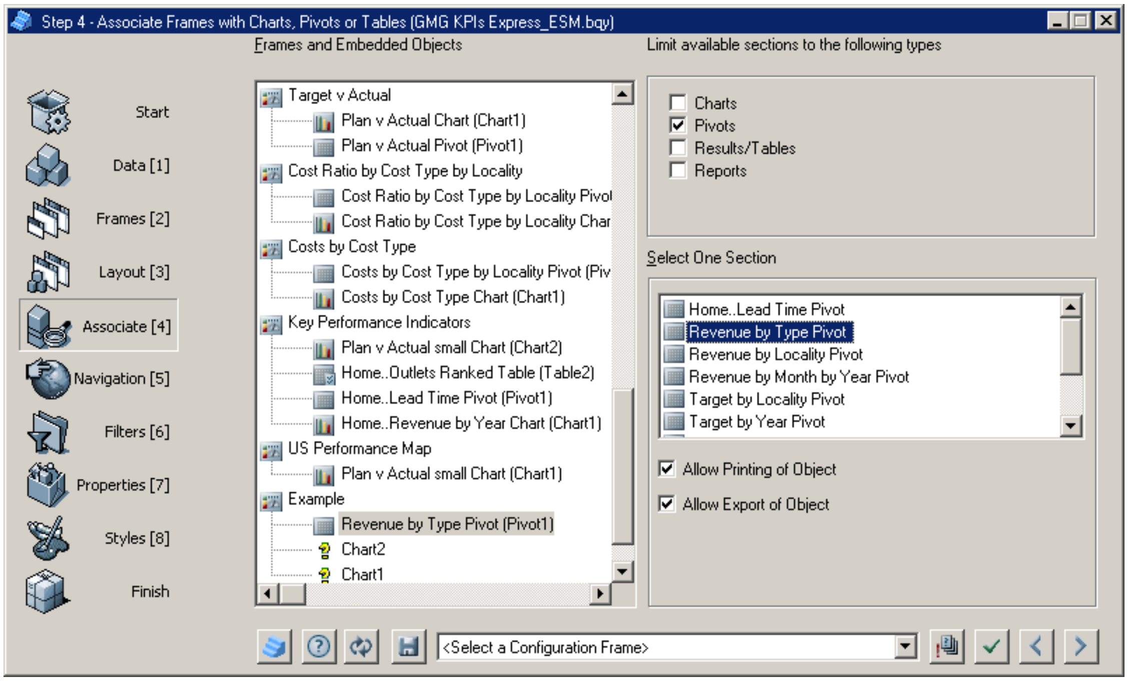
Using Controls for Embedded Objects

The print and export object controls are selected by default.

- **Allow Printing of Object**—Enables printing of selected embedded objects (To add reports to the list of printable objects, see "Configuring and Embedding Reports" on page 227)
- **Allow Export of Object**—Enables export of selected embedded objects

➤ To use the controls for embedded objects:

1 In **Step 4: Associate Frames with Charts, Pivots, or Tables**, from **Frames and Embedded Objects**, select an embedded object.



Frames and Embedded Objects

- Target v Actual
  - Plan v Actual Chart (Chart1)
  - Plan v Actual Pivot (Pivot1)
- Cost Ratio by Cost Type by Locality
  - Cost Ratio by Cost Type by Locality Pivot
  - Cost Ratio by Cost Type by Locality Char
- Costs by Cost Type
  - Costs by Cost Type by Locality Pivot (Piv
  - Costs by Cost Type Chart (Chart1)
- Key Performance Indicators
  - Plan v Actual small Chart (Chart2)
  - Home..Outlets Ranked Table (Table2)
  - Home..Lead Time Pivot (Pivot1)
  - Home..Revenue by Year Chart (Chart1)
- US Performance Map
  - Plan v Actual small Chart (Chart1)
- Example
  - Revenue by Type Pivot (Pivot1)
  - Chart2
  - Chart1

Limit available sections to the following types

☐ Charts
☑ Pivots
☐ Results/Tables
☐ Reports

Select One Section

- Home..Lead Time Pivot
- Revenue by Type Pivot
- Revenue by Locality Pivot
- Revenue by Month by Year Pivot
- Target by Locality Pivot
- Target by Year Pivot

☑ Allow Printing of Object

☑ Allow Export of Object

<Select a Configuration Frame>

2 **Clear the print or export options.**

These options are selected by default.

3 **After the object controls are configured, click ✓ to apply the selections to the document.**

4 **Click > to move to the next step.**

Step 5: Configure Navigation is displayed.

Step 5: Configure Navigation

This step enables you to specify the display on the side-label, drop-down list, or tab navigation of the dashboard. Navigation targets enable you to move through all predefined views of a dashboard created with Dashboard Studio.

A flat navigation structure (identical on every page) is the easiest for end users to follow. However, a hierarchy of navigation targets can be built by enabling multiple targets to be available for each frame. Use of a hierarchy enables you to accommodate more targets in a dashboard.

Note:

The navigation step includes only the frames that do not contain grouped tabs for navigation. If all frames include navigational grouped tabs, then Step 5 is blank. See the “Dashboard Studio Group Tabs Component” chapter in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

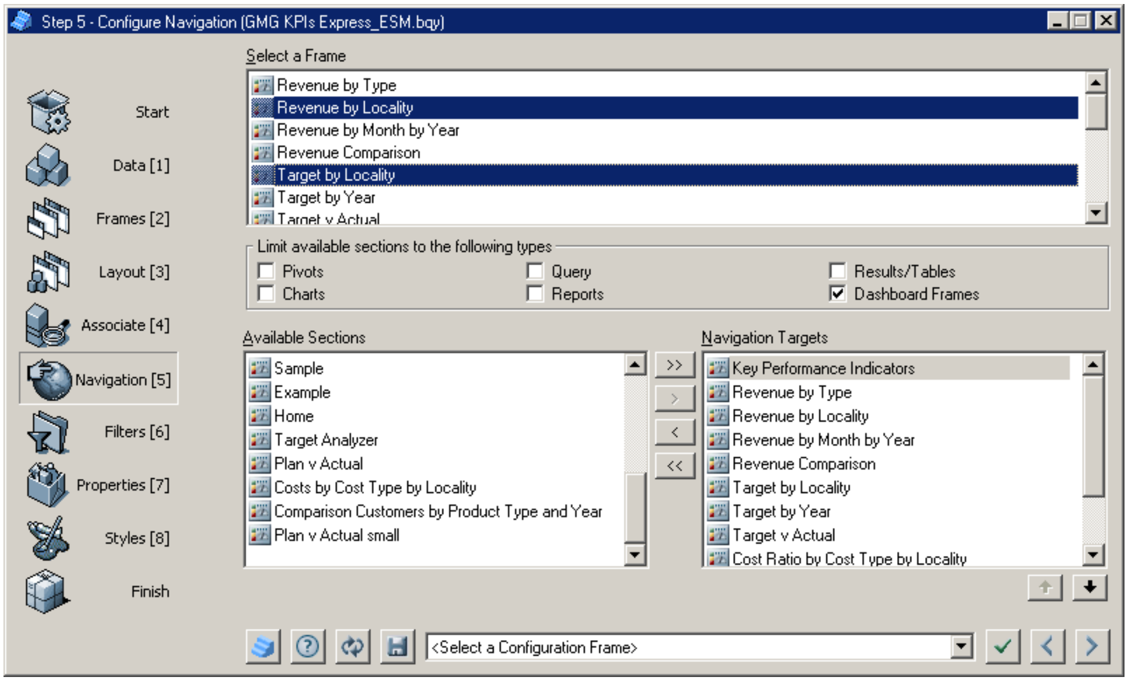
No limit exists as to the number of sections that can be created; however, a limit exists as to the number of targets that can be added to a frame.

Tip:

A dashboard that contains many sections may become cumbersome. Consider dividing such a dashboard into two or more dashboards that are linked through the Dashboard Studio Related Document feature. See "Opening Related Documents" on page 32.

➤ To configure navigation:

1 **In Step 5: Configure Navigation from Select a Frame, select one or more frames.**



Select a Frame

- Revenue by Type
- Revenue by Locality
- Revenue by Month by Year
- Revenue Comparison
- Target by Locality
- Target by Year
- Target v Actual

Limit available sections to the following types

□ Pivots □ Query □ Results/Tables
□ Charts □ Reports ☑ Dashboard Frames

Available Sections

- Sample
- Example
- Home
- Target Analyzer
- Plan v Actual
- Costs by Cost Type by Locality
- Comparison Customers by Product Type and Year
- Plan v Actual small

Navigation Targets

- Key Performance Indicators
- Revenue by Type
- Revenue by Locality
- Revenue by Month by Year
- Revenue Comparison
- Target by Locality
- Target by Year
- Target v Actual
- Cost Ratio by Cost Type by Locality

2 **Optional:** To display a report or a non-dashboard section on a dashboard, in **Limit available sections to the following types**, select the section type.

The selected sections are displayed in Available Sections.

3 Move selected sections between **Available Sections** and **Navigation Targets** by clicking > and <, or move all sections by clicking >> and <<, or double-click a section.

4 **Optional:** Reorder **Navigation Targets,** by selecting a target and clicking ↑ or ↓.

5 After navigation is configured, click ✓ to apply the selections to the document.

6 Click > to move to the next step.

Step 6: Configure Filters is displayed.

Step 6: Configure Filters

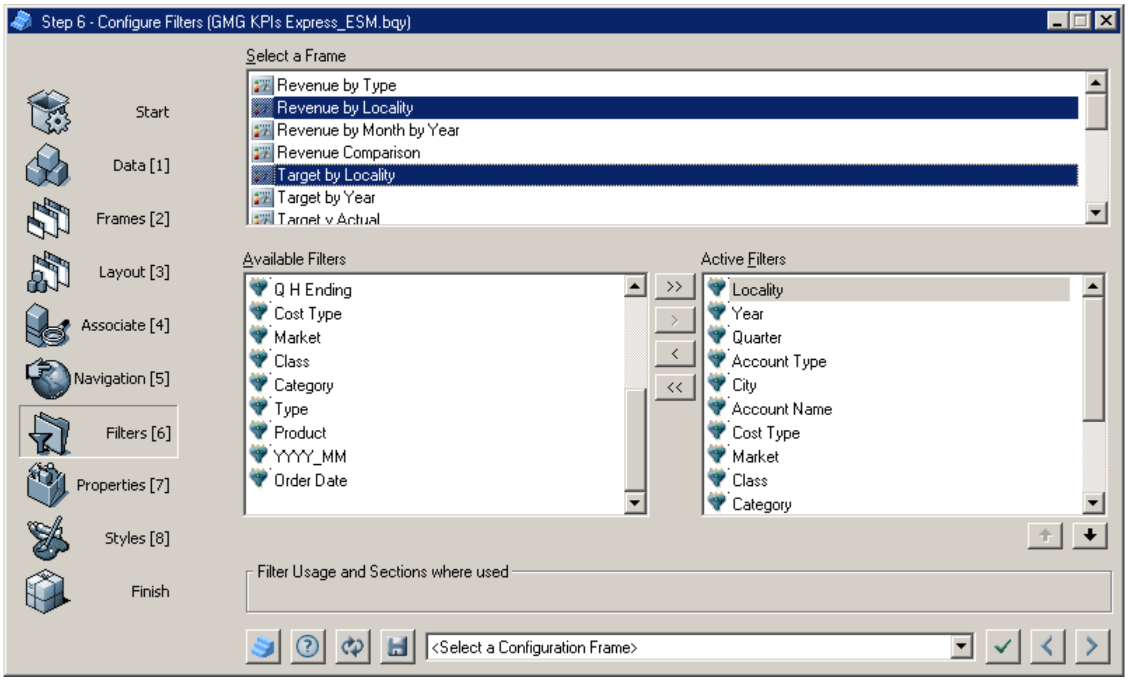
This step enables you to set filters that are available for each dashboard section. The filters are available in the Filters drop-down list on the top panel of the dashboard.

Note:

Lite frame prototypes cannot directly access the Filter dialog box, so frames created from Lite frames are excluded from the list.

➤ To configure filters:

1 **In Step 6: Configure Filters from Select a Frame, select one or more frames.**



2 **Optional: To add or remove filters from the dashboard, use the Configuration drop-down list to move to QIQ Filter Properties, and click ⟳ to refresh settings.**

3 **Select Available Filters to be available in the selected dashboard sections.**

Available Filters only displays filters created before arriving at Step 6 of Dashboard Studio.

Note:

Filter names that start with _sys (or with sys for pre-3.2 Release templates) are ignored by the filters feature of Dashboard Studio and are not altered in any way.

4 **Move selected filters between Available Filters and Active Filters by clicking > and <, or move all filters by clicking >> and <<, or double-click a filter.**

5 **Optional: Reorder Active Filters, by selecting a filter and clicking ↑ or ↓.**

6 Click ✓ to apply the selections to the document.

7 Click > to move to the next step.

Step 7: Configure Properties is displayed.

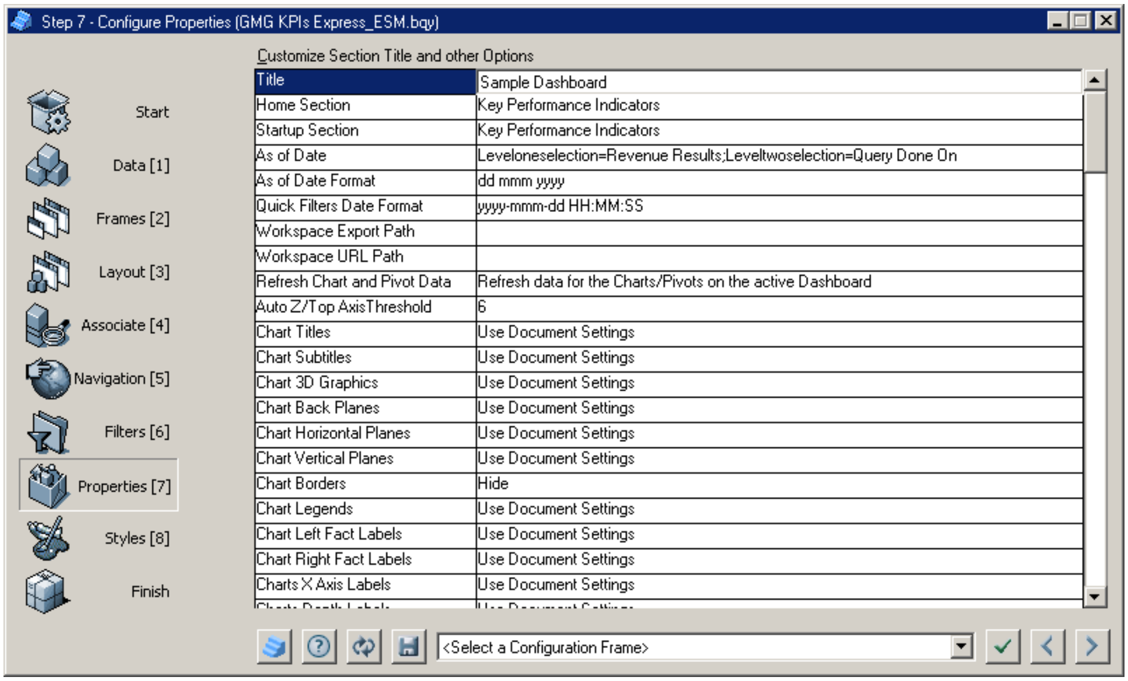
Filter Usage and Sections where used indicates what is represented by the currently selected filter. For example, if selected filters are in Revenue Results, filters that share a name and data type are treated as one filter. A combined set of filters is presented, and the selected values are distributed to all results with filters.

Caution!

If a column is displayed in multiple tables or results with an identical name but conflicting data types, unexpected behavior may result in the dashboard. Such columns must be renamed so that each column of each filter contains matching data types.

Step 7: Configure Properties

This step enables you to set dashboard properties. All properties are optional.



Customize Section Title and other Options

Title	Sample Dashboard
Home Section	Key Performance Indicators
Startup Section	Key Performance Indicators
As of Date	Leveloneselection=Revenue Results;Leveltwoselection=Query Done On
As of Date Format	dd mmm yyyy
Quick Filters Date Format	yyyy-mmm-dd HH:MM:SS
Workspace Export Path	
Workspace URL Path	
Refresh Chart and Pivot Data	Refresh data for the Charts/Pivots on the active Dashboard
Auto Z/Top AxisThreshold	6
Chart Titles	Use Document Settings
Chart Subtitles	Use Document Settings
Chart 3D Graphics	Use Document Settings
Chart Back Planes	Use Document Settings
Chart Horizontal Planes	Use Document Settings
Chart Vertical Planes	Use Document Settings
Chart Borders	Hide
Chart Legends	Use Document Settings
Chart Left Fact Labels	Use Document Settings
Chart Right Fact Labels	Use Document Settings
Charts X Axis Labels	Use Document Settings
Charts Depth Labels	Use Document Settings

These are the main property areas that are available for configuration in the step.

- **Title**—Enter a title, which is also the default file name for the dashboard (The title is displayed on the Home frame that is created from frame prototypes that support the Title property)
- **Home Section**—Specify a Home frame for the dashboard (Available in only some frame prototypes. See “Specifying a Home Section” on page 77)

- **Startup Section**—Specify the first frame that is displayed when the dashboard is opened (All available sections are listed, including frames not created with Dashboard Studio. The Filter Control EIS frame is available)
- **As of Date**—Select a date column that must contain date as the data type (As of Date, identifies the timeliness of the dashboard data. See "Defining the As of Date Column" on page 52. If the column contains multiple values, the first date column value is selected)
- **As of Date Format**—Specify the format in which the As of Date is displayed (Multiple formats are available)
- **Quick Filters Date Format**—Specify the default format of the date filters in the dashboard (By default, date filters without formats assigned to them in QIQ Filter Properties use the yyyy-mmm-dd hh:mm:ss format. Date formats set in QIQ Filter Properties take precedence over this property. The default format is applied if an invalid date format is specified in this step or QIQ Filter Properties)
- **EPM Workspace Export Path**—Specify the server location where files are saved when sections are exported (See "EPM Workspace Export Path" on page 77)
- **EPM Workspace URL Path**—Specify the Web browsers path used to access exported files (See "EPM Workspace URL Path" on page 77)
- **Refresh Chart and Pivot Data**—Set to control how charts and pivots are updated when filters are set or a query is reprocessed (Refresh all data simultaneously or when the data is moved to the frame. Select the Use Document Settings command if this property is not to be set by Dashboard Studio)
- **Auto Z/Top Axis Threshold**—Set the threshold for the Auto Z/Top Axis feature that was applied in "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68 (The default threshold is 4. If too many values are displayed, a chart or pivot is difficult to read)
- **Chart Settings**—Set to manipulate the format and behavior of charts embedded within frames (Select the Use Document Settings command if chart options are not to be set by Dashboard Studio)
- **Toolbars, Menus, Status Bar, and Title Bar Settings**—Set to display or hide toolbars, menus, the status bar, and the title bar in the plug-in and desktop editions of Interactive Reporting Studio (Select True for these properties to set the features to visible. Plug-in features apply in a Web browser or thin-client environment. LAN client features apply in a client environment. Only runtime dashboards are affected by these properties. ESM dashboards open with typical Interactive Reporting Studio settings)

➤ To configure properties:

1 **In Step 7: Configure Properties, change one or more property settings.**

An example is to change Chart Titles by removing object types. If a prefix is used to group multiple charts on a frame, changing the chart title removes the prefix. That is, if a chart is named *Home..Total Sales chart*, the title is changed to *Total Sales*.

2 **Click ✓ to apply the selections to the document.**

3 **Click > to move to the next step.**

Step 8: Configure Style is displayed.

EPM Workspace Export Path

The location must be a directory that the server can access. This path generally points to a folder in the server directory; for example, `C:\Program Files\Brio\Brio8\Servlets\Deployment`. This property is required if a dashboard is imported into iServer (EPM Workspace) Release 8.1 or earlier, and you export a section by clicking Export on the dashboard toolbar. When running dashboards in the EPM Workspace thin-client, only the thin-client standard toolbar preference, including the paging toolbar, is available.

EPM Workspace URL Path

The path is directly related to the specified EPM Workspace Export Path. For example, the Home section exported from a dashboard that is imported into EPM Workspace, is saved as: `C:\Program Files\Brio\Brio8\Servlets\Deployment\Home.pdf`. EPM Workspace uses a URL path of the form: `http://<your server>:<your port number>/Brio/`.

For example, assume that your EPM Workspace home page is `http://www.myServer.com:8080/Brio/browse/Main` and the URL path is `http://www.myServer.com:8080/Brio/`, when Export is clicked, your Web browser reads the exported file from the URL: `http://www.myServer.com:8080/Brio/Home.pdf`.

This property is required if a dashboard is imported into iServer (EPM Workspace) Release 8.1 or earlier . When running dashboards in the EPM Workspace thin-client, only the thin-client standard toolbar preference, including the paging toolbar, is available

Specifying a Home Section

A set of rules defines the Home section (first displayed section seen upon opening a document).

- The section is explicitly defined through the object model; for example, `Activate()`
- The section is explicitly selected on the Dashboard Home dialog box
- If the section is not explicitly defined through the object model or the user interface, the section that was last saved is opened first
- If Home is not set, then the last saved section is opened (If the user is cannot access the last saved section; for example, due to an adaptive state which prevents a user from viewing a section, then the document is opened in a section available to the user)

In Interactive Reporting Studio , there is no default Home section, that is, if a section is not explicitly defined, none is assumed in Interactive Reporting Studio.

Using the HomeDashboard Property

The creation of the HomeDashboard property in ActiveDocument means any dashboard section can be specified as the Home section using the object model. Using the HomeDashboard property in Interactive Reporting Studio produces equivalent results to using the Dashboard Home dialog box, or clicking on the toolbar when in Design mode, or setting a Home section in Step 7: Configure Properties of Dashboard Studio.

You can modify the example to suit your requirements.

Note:

When specifying a Home section in Dashboard Studio, if the is invisible, enter Design mode and select Dashboard, then Home Dialog on the Interactive Reporting Studio menu bar to activate and the Dashboard Home dialog box.

➤ To use the HomeDashboard property:

1 **In Interactive Reporting Studio, select File, then New.**

New File is displayed.

2 **Select Other, then A Blank Document, and then click OK.**

The document opens in Design mode.

3 **From Sections, right-click Dashboard, and select Duplicate Section.**

4 **Right-click Dashboard2, and select Rename Section.**

For example, enter `First Dashboard`.

5 **Perform an action:**

- Select **Dashboard**, then **Home Dialog**
- Click

Dashboard Home is displayed listing Dashboard and First Dashboard.

6 **Select Dashboard (the original section) and click OK.**

7 **Press Ctrl+D to exit Design mode, and click .**

The active section becomes Dashboard (if not already selected).

8 **Press Ctrl+D to return to Design mode.**

9 **From the catalog pane, expand Controls, and drag a command button onto Dashboard.**

10 **Right-click CommandButton1, and select Scripts.**

Script Editor is displayed.

11 **Copy and paste this script, or modify to suit your requirements, and click OK.**

```
ActiveDocument.HomeDashboard = "First Dashboard";
```

12 **Press Ctrl+D to exit Design mode, and click CommandButton1, and then click .**

The active section becomes First Dashboard, demonstrating that the Home dashboard has changed from Dashboard to First Dashboard.

13 **From Sections, select Dashboard.**

14 **Save and close the dashboard.**

15 **Reopen the dashboard.**

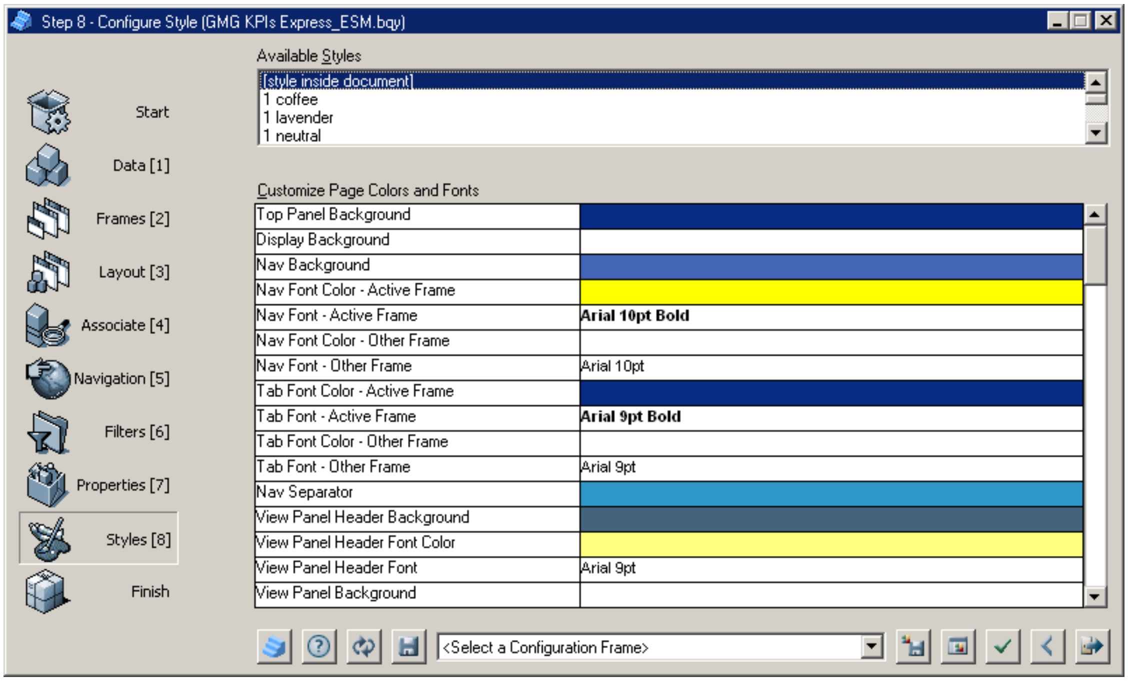
First Dashboard is displayed as the Home section for the dashboard.

Pre-Load Active Dashboard Home Sections

Interactive Reporting Studio users can use the Dashboard Home dialog box to enable the pre-loading of an active dashboard Home section in order to increase loading performance in EPM Workspace. This option is enabled only when the selected dashboard section in an Interactive Reporting document contains a minimum of one dashboard with one embedded browser.

Step 8: Configure Style

This step enables you to set the style of dashboard elements. A style describes the color scheme and text characteristics of the frame.



Step 8 - Configure Style (GMG KPIs Express_ESM.bqy)

Available Styles

[style inside document]
1 coffee
1 lavender
1 neutral

Customize Page Colors and Fonts

Top Panel Background	🟦
Display Background	⬜
Nav Background	🟦
Nav Font Color - Active Frame	🟨
Nav Font - Active Frame	**Arial 10pt Bold**
Nav Font Color - Other Frame	⬜
Nav Font - Other Frame	Arial 10pt
Tab Font Color - Active Frame	🟦
Tab Font - Active Frame	**Arial 9pt Bold**
Tab Font Color - Other Frame	⬜
Tab Font - Other Frame	Arial 9pt
Nav Separator	🟦
View Panel Header Background	🟦
View Panel Header Font Color	🟨
View Panel Header Font	Arial 9pt
View Panel Background	⬜

<Select a Configuration Frame>

Style Elements

A style is a collection of formatting specifications that can be applied to the dashboard or to individual elements. Style elements are located in Customize Page Colors and Fonts and include these items.

- **Top Panel Background**—Dashboard buttons, company logo (usually on left side), and so on
- **Display Background**—Dashboard background color
- **Nav Background**—Panel on the left (contains the hyperlinks)
- **Nav text elements**—Text displayed on the navigation bar (divided into text for the active frame and others)
- **Nav Separator**—Line that separates the hyperlinks on the navigation bar
- **Tab text elements**—Text displayed on the navigation tabs (divided into text for the active frame and others)
- **View Panel Background and Header elements**—Background, font color and style
- **Quick Filters, Active Filters, and Quick Slice text elements**—Check box text and option button filter values
- **Quick Filters, Active Filters, and Quick Slice Background**—Background color
- **Quick Filters, Active Filters, and Quick Slice Headings**—Heading displayed for Quick Filters, Active Filters, and Quick Slice
- **Title Text elements**—Title text presented on the background
- **As of elements**—As of date text, (shows the timeliness of the data)
- **Tool Tip elements**—Text displayed when hovering over a button, or Help is clicked

Changing Styles

Available Styles displays a list of predefined styles.

Note:

Styles that have a '1' prefix are designed for Webdash frames.

➤ To apply a style:

1 In **Step 8: Configure Style** from **Available Styles**, select a style.

2 Click ✓ to apply the selection to the document.

The style elements on the current dashboard are set to the values specified in the selected style.

➤ To manually configure each style element:

1 In **Step 8: Configure Style** from **Customize Page Colors and Fonts**, select a style element.

- If a color element is selected, the Color dialog box is displayed
- If a font element is selected, the Font dialog box is displayed

2 If a color element is selected, in the **Color** dialog box, complete one of these actions:

- From the Basic colors table, select a color
- From the Custom colors table, select a color
- On the right of the Color dialog box, set RGB values

3 If is clicked, the palettes folder is opened; from the palettes folder, select a predefined color scheme (a PAL file); and click **Open**.

A dialog box is displayed explaining that the selected palette is successfully loaded. The palette is available as a set of colors, in the Custom colors table of the Color dialog box.

4 If a font element is selected, in the **Font** dialog box, make the modifications, and click **OK**.

5 Click ✓ to apply the selections to the document, and view the changes in the dashboard.

6 After setting style elements, click to save the style.

7 In **Save As**, save the style with an FDS file extension.

For example, Basic.fds. The style is displayed in Available Styles and can be used in other dashboards.

8 Click to save the current dashboard without closing it.

9 If is clicked, in **Save As**, accept the default file name or specify another name.

If the specified name exists, save the file with another name.

If the default file name is accepted, an *_esm* suffix is added to the file name. If a file name is specified, the *_esm* suffix is not added.

10 After naming the dashboard, click **Yes**, when a message is displayed asking to save a slimmer, runtime version of the dashboard.

Select a Framework Template is displayed.

11 Alternatively, if the document is closed and a runtime version is saved (using a template Release 8.3.8 or later), select **Pack JavaScript** to compact the JavaScript and remove all trace statements.

See "Packing JavaScript Feature" on page 190.

New dashboards are saved to the default save location associated with the template from which they were built. Runtime dashboards are saved to a Runtime folder in the location of the saved ESM. Existing dashboards are saved to the location from which they were opened.

Configuration Frames

Dashboard Studio templates, Release 8.2 and later, provide configuration frames. These frames are accessible from Dashboard Studio when a dashboard is created, Interactive Reporting Studio Sections catalog, or the Configuration drop-down list of Dashboard Studio. Each configuration frame includes a navigation drop-down list on the top panel, that enables you to access other available dashboard configuration frames.

A set of buttons (|<, <, >, >|) is included on configuration frames to enable easy navigation from one step to another in a multi-form configurator. For single-form configurators, these buttons are not selectable.

The standard template contains these configuration frames:

- QIQ Filter Properties
- Filter Enhancements
- QIQ Related Document Properties
- Wide View Properties
- Pivot Fact Properties
- QIQ Sort Properties
- List Expansion Properties
- Language Properties
- Date Formats
- Color Palette Properties
- Chart Spotlight Properties (see “Spotlighting the Chart” on page 200)
- Create Summary Table (see “Creating Summary Tables” on page 198)
- The Webdash template includes a configuration frame called *Build Dashboard*
- Layout Manager Properties

Note:

Because dashboards created with Dashboard Studio are built from a customizable, extensible framework, configuration frames may exist that are not documented in this guide. If a configuration frame is part of a Dashboard Studio component, it is documented in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

Configuration frame availability varies depending on the state of the dashboard. Frames are part of the design process and are available only in templates and ESMs; therefore they are not available in runtime versions.

Using the QIQ Filter Properties Frame

QIQ Filter Properties enables you to create local filters, set a format for the date filters, and specify startup filters.

Creating Filters

Note:

The Local Filters section is only available on QIQ Filter Properties if Webdash is *not* available in the Interactive Reporting document. Where Webdash is available, create the local filters using the Build Dashboard frame. See "Creating Local Filters" on page 100.

Under Local Filters you can create local filters for the dashboard.

Note:

Filter names that start with _sys (or with sys for pre-3.2 Release templates) are ignored by the filters feature of Dashboard Studio and are not altered in any way.

➤ To create local filters:

1. **In Interactive Reporting Studio, navigate to QIQ Filter Properties.**

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. **From the Local Filters drop-down list, select a table or results that contains the columns to be set as local filters.**

   Columns lists all available columns from the selected table or results. Table or results that are saved without results are not displayed in this list until they are processed.

3. **Move selected filters between Columns and Local Filters by clicking > and <, and click Set Filters.**

   The selected columns are set as local filters in the table or results and are available in the Filters step of Dashboard Studio. See "Step 6: Configure Filters" on page 74.

   If a column is displayed in multiple tables or results with an identical name but conflicting data types, unexpected behavior may result in the dashboard. Such columns must be renamed so that each column of each filter contains matching data types.

4. **Repeat steps 2–3, for other table or results that contain columns to be set as local filters.**

   Computed item columns are not displayed in the Columns list. Creating local filters including computed item columns requires further steps.

➤ To create local filters including computed item columns:

1. In Interactive Reporting Studio, navigate to the table or results that contains the computed item column.
2. Double-click the computed item column.

   The Filter dialog box is displayed.

3. Click **Select All** and click **OK**.
4. To refresh **Columns** in QIQ Filter Properties, preform an action:
   - From the Local Filters drop-down list in QIQ Filter Properties, select the table or results that contains the computed item column, and click Set Filter
   - From Select a Frame in Step 6: Configure Filters of Dashboard Studio, select the frame that contains the table or results with the computed item column, and click ✓

   In QIQ Filter Properties, Columns is refreshed and the computed item column is displayed as an available column.

Using Cascading Filters

QIQ Filter Properties enables you to specify whether filters set in the dashboard are cascading, which is the default. For example, if locality and city filters exist, setting the locality filter refreshes the list of city values, so only cities within the selected locality are available.

However, setting the filters to be non-cascading makes the dashboard more efficient. If filters are non-cascading, lists do not need to be refreshed so frames may activate more quickly.

Tip:

It is recommended that the cascading filters property is enabled during the dashboard building process because some properties on the frame are not refreshed when the cascade property is turned off. Turn the cascade property off after the dashboard is created.

➤ To use cascading filters:

1. In Interactive Reporting Studio, navigate to **QIQ Filter Properties**.

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. From **Cascading Filters**, select **On**.
3. Click **Set** to apply the selection.

   The Set process may be time-consuming if the results contain many rows. If performance issues are a concern, select On without clicking Set. The selection is applied the next time the dashboard is opened.

   Each time a query is processed, performance is slow initially, as the filters are applied or removed. If the cascading filters feature is turned off, the dashboard applies filters and activates frames more efficiently.

Tip:

It is not recommended that the cascading feature is turned off if a date filter whose HH:MM:SS AM timestamp is not 12:00:00 AM.

Note:

When an Interactive Reporting document is processed with Scheduler and the selected cascading command is Off, the available values are refreshed in all Quick Filters, resulting in the Interactive Reporting document being faster to open.

Setting Filter Formats

Filters Format of QIQ Filter Properties enables you to specify a format for date filters in the dashboard.

➤ To specify filter formats:

1. **In Interactive Reporting Studio, navigate to QIQ Filter Properties.**

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. **Select a format.**

   Date formats must be entered using the format, y = year, m = month, d = day, H = hour, M = minute, and S = second. Apart from separators, no other alphanumeric characters are accepted. Separators must be included between date elements. For example, *mmddyy* is not a valid format.

   Date filters cannot include “>” as part of the name, because the character is used as a separator by Dashboard Studio, and if used, an error is generated.

   A list of all date filters in the dashboard is displayed in Date Filters.

3. **Select a date filter, and click > to assign a format to the filter.**

4. **Click Set Formats.**

   Formats specified in QIQ Filter Properties take precedence over the default format specified in “Step 7: Configure Properties” on page 75.

Setting Starting Filters

Under Starting Filter Values you can configure a set of filter values that is applied when the dashboard is first opened.

➤ To specify starting filters:

1. **In Interactive Reporting Studio, navigate to QIQ Filter Properties.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **In Starting Filter Values, from the Filters drop-down list, which contains a list of all local filters set for the dashboard, select a filter.**

The Filter dialog box is displayed.

3 **Perform an action:**

- If the operator is set to a value other than Is Null, select the filter values to be set, and click OK
- If the operator is set to Is Null, select a value, and click OK

The selected values are automatically set as starting filters.

4 **Optional: To remove a starting filter, select a filter, and click Remove Selected.**

Note:

If Remember filter settings on startup is activated from Filter Control EIS, the filters set when the dashboard was closed take precedence over the startup filters.

Tip:

The startup filters are set as the starting checkpoint. See "Using Checkpoint Control" on page 32.

Using the Filter Enhancements Properties Frames

Filter enhancements relate to templates and include two properties frames; Filter Aliases and Filter Exclusions.

Note:

The Filter Enhancement feature means that ESM and runtime documents are incompatible with the Move Filters feature of Dashboard Studio Optimize Utility in Release 9.3 and earlier. See "Moving Filters Feature" on page 189.

Filter Aliases Properties Frame

In releases earlier than Release 9.3.1, all local filters with an identical display name are logically treated as if they are one filter with a set of combined values. However, there are cases where filters with different names should be treated as one, and filters with the same display name should be treated separately.

The Filter Aliases feature is backward compatible with existing dashboards. When a document is updated, all filters are converted to the alias format. By default the alias is the display name of the filter. Use the properties frame to change the default if is it is not the required name. By using the properties frame, you can create, delete, and rename aliases, and combine columns from different sections with different names, and separate columns with the same name.

Note:

Clicking Default reverts your settings to the filter default rule in the collection.

Local filters are treated as one filter:

- Where they share a set of selectable values in the results section that they filter
- When applying identical selections to all sections in one action

Local filters are treated as different filters:

- Where they have different selectable values and selections cause results to display no data
- When applying the selections only to a subset of sections in one action

➤ To use the Filter Aliases properties frame:

1. **In Interactive Reporting Studio, navigate to Filter Aliases.**

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. **In Filter Alias Name, enter an alias.**

   Alias names are displayed.

3. **To add the filter alias name to the Alias Name list, click .**
4. **Select an Alias Name.**
5. **Optional: To rename the selected alias, click .**
6. **Optional: To delete and return the selected alias to Unsettable Filters, click .**
7. **Optional: Reorder Alias Name entries by selecting an alias and clicking ˄ or ˅.**
8. **Move selected filters between Unsettable Filters and Aliased Filters by clicking > and <.**
9. **Perform an action:**
   - To revert your settings to the filter default rule, click Default
   - To discard selected filter aliases and reload the original settings, click Restore
   - To remove all entries, click Clear All
   - To remove selected filters, click Clear
   - To load selected filter aliases, click Apply

Note:

Filter Aliases are recognized in the Dashboard Studio Optimize Utility when a section is renamed.

Filter Exclusion Properties Frame

In earlier releases, filter status is displayed on every dashboard, and filters selection is performed in Step 6: Configure Filters of the Dashboard Studio wizard. See "Step 6: Configure Filters" on page 74. The earlier default methods are effective where there is only one results set that provides data to visualizations (charts, pivots, spotlight shape sets, and graphic gauges) or where there are multiple results sets with the same filters and common values.

If you change a filter setting on a dashboard but no changes are reflected in any of the views, then the visualizations must be based on different results sets which are displayed on different dashboards. Use the Filter Exclusion properties frame to analyze the document dependencies, and to ensure filters are correctly deployed and displayed on dashboards. Click Default, and select a Filter drop-down setting option.

The Filter Exclusion feature treats graphic gauges (see "Dashboard Studio Graphic Gauges Component" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*) and Chart Spotlight shape sets (see Chapter 12, "Spotlighting Charts") as visualizations and analyzes them to decide which to include or exclude.

Note:

The Filter Exclusion feature is backward compatible with current functionality. The default is to exclude nothing and to support pre-9.3.1 Release behavior.

Example: Using the Filter Exclusion Feature

As an example of the Filter Exclusion feature, use an Interactive Reporting document with three dashboards (Home, Sales, and Customers) and two results sets. One results set contains sales and service revenue information, and the other contains customer satisfaction survey data.

Set filters to apply to both results sets; for example, year, quarter, and customer type. Set other filters; for example, city, sales group, sales representative, and product type to apply only to sales information. And apply filters; for example, type of use and age only to the customer satisfaction data.

Ensure the dashboards display visualizations and information reflected in the results sets. For example, the Home dashboard displays revenue and satisfaction visualizations, the Sales section displays only revenue data, and the Customers dashboard displays only satisfaction information.

Using the Filter Exclusions properties frame, click Default to perform these actions:

- On Home, no exclusions are set, as all filters can affect visualizations
- On Sales, the filters: type of use and age are excluded, as these apply only to customer satisfaction

- On Customers, the filters: city, sales group, sales representative, and product type are excluded, as these do not apply to satisfaction information

➤ To use the Filter Exclusions properties frame:

1 **In Interactive Reporting Studio, navigate to Filter Exclusions.**

2 **From Filter drop-down setting, select an option to configure filter drop-down lists on dashboard frames and automatically assign filters to frames:**

   - Add and remove filters from Filter drop-down lists—Automatically add filters that affect a dashboard, and remove filters that do not affect the frame (This option overwrites current filter configurations)
   - Remove excluded filters from Filter drop-down lists—Remove filters that do not affect the dashboard frame, and do not add filters that are not already added)
   - Do nothing to the Filter drop-down lists—Filter configurations are not touched and no automatic filter assignments are added

3 **To apply a selected Filter drop-down setting, click Default.**

4 **From Dashboards, select a section manual exclusions to filters.**

   Available Filters is populated.

5 **Optional: To view the section, click 🗔.**

6 **To copy excluded filters from the selected section to memory, click ⧉.**

7 **Optional: To paste excluded filters from memory into the selected section, click 📋.**

   📋 is visible only if you have clicked ⧉.

8 **Move selected filters between Available Filters and Filters excluded from displays by clicking > and <.**

9 **Perform an action:**

   - To discard selected filter and reload the original settings, click Restore
   - To remove all entries, click Clear All
   - To remove selected filters, click Clear
   - To exclude the selected filters, click Apply

Applying Filter Enhancement Properties

For the most effective application of Filter Aliases and Filter Exclusions, perform these steps in order:

1. Using the QIQ Filters properties frame, create filters (see“Using the QIQ Filter Properties Frame” on page 83).
2. **Optional:** Using the Filter Aliases properties frame, create filter aliases.
3. Using Step 6: Configure Filters of the Dashboard Studio wizard, deploy filters (see “Step 6: Configure Filters” on page 74).
4. **Optional:** On the Filter Exclusions properties frame, click **Default**, to remove exclusions.

Alternatively, perform these steps in order:

1. Using the QIQ Filters properties frame, create filters (see“Using the QIQ Filter Properties Frame” on page 83).
2. **Optional:** Using the Filter Aliases properties frame, create filter aliases.
3. On the Filter Exclusions properties frame, perform these actions:
   - To remove exclusions and add available filters, click Default
   - To save the changes to the dashboard frame, click Apply

Using the QIQ Related Document Properties Frame

From QIQ Related Document Properties, you can set predefined, related documents that are available from all frames. Related documents can be files, folders, URLs, or e-mail messages that are launched from within the dashboard.

➤ To use the related documents function:

1 In Interactive Reporting Studio, navigate to QIQ Related Document Properties.

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 In Display Name, enter a title for the related document.

The name that is displayed in the Related Documents drop-down list when is clicked on the dashboard top panel.

3 In Document Path, enter the file path, folder location, URL, or e-mail address to associate with the specified title.

These syntax rules apply.

File or Folder—A universal naming convention (UNC) shared file or folder location can be entered to bring up a file or folder from a shared location. It is recommended that folders are accessed through Network Neighborhood or My Network Places as these systems generate UNC names that work from a computer on a LAN.

URL—Text beginning with a standard Internet protocol prefix (`http://`, `ftp://`, www, and so on), opens the default Web browser. For example, if www.hyperion.com is entered, clicking (Related Document) on the dashboard launches the Oracle Web site. This functionality can also be used to link to Interactive Reporting documents available through Oracle Hyperion Enterprise Performance Management System On-Demand Server.

E-mail—Text that is prefixed with `mailto:` brings up the registered e-mail client. For example, if `mailto:support@`*`company`*`.com` is entered, clicking on the dashboard launches the e-mail client and creates an e-mail pre-filled with the specified e-mail address on the `To:` line. The syntax for the `mailto:` URL conforms to rfc2368. Search for rfc2368 in a Web search engine for all syntax rules and information about how to pre-fill the subject line or the body of the e-mail.

The ability to activate folders, URLs, or e-mail clients works only if the dashboard is viewed in Windows. Non-Windows users are able only to specify a file as the related document, so the path to the handler must be entered explicitly, because Dashboard Studio works exclusively in Windows.

4 Click **Browse** (next to Document Path) and locate a file or folder to be a related document.

As a result of necessary components, such as Visual Studio or Visual Basic, not being installed correctly on the computer, Browse may not be visible on the frame. If Dashboard Studio detects that Browse may not work properly, it hides it from view.

5 **Optional:** To use another program other than the default to open the related document, in **Document Handler**, enter the path of the application.

For example, to open the document in Word, locate `Winword.exe`.

If a document handler is explicitly set, to view the related document, a user must have the handler installed in the set location.

Some document handlers (such as Interactive Reporting Studio `brioqry.exe`) do not recognize the `file:///` document path format. To specify a document handler that does not recognize the document path, change the document path to a standard Windows path.

For example, change `file:///C:/Related Document.bqy` to read `C:\Related Document.bqy`.

The `file:///` item enables the related document to be launched when the dashboard is imported into EPM Workspace. However, changing to a standard Windows path may prevent the document opening in EPM Workspace.

6 After the related document is configured, click **Test** to launch the document and check the settings.

7 Click **Add** to insert the configured related document.

8 **Optional:** Reorder **Related Documents List** items, by selecting an item and clicking ▲ or ▼.

9 **Optional:** To delete the document from the list, select it, and click **Remove**.

Wide View Properties

Wide View Properties enables you to specify shapes that are visible when a dashboard is switched between Wide View and Standard View modes.

Tip:

Configure Quick Filters and Quick Slice objects *before* adding Wide View features because it is difficult to modify the objects later.

Note:

Ensure that charts and pivots are configured *before* adding Wide View features. See "Configuring Wide View Mode for Charts and Pivots" on page 57.

Using the Wide View Properties Frame

➤ To configure dashboard shapes:

1 **In Interactive Reporting Studio, navigate to Wide View Properties.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **From Dashboard Sections, select an option to display the list of dashboard sections.**

- All dashboards—All frames are available
- Configured dashboards—Only frames configured with Wide View mode are available
- Unconfigured dashboards—Only frames that are not configured with Wide View mode are available

3 **From the list, directly below the options, select a section.**

Available Shapes are displayed.

- To view the selected section, click
- To copy a list of shapes configured to be hidden for one frame, click
- To paste a list of shapes onto another frame, click

Shapes are only copied provided they exist in the sections on which they are pasted.

4 **Move selected shapes between Available Shapes and Shapes Hidden in Wide Mode by clicking > and <.**

Selected shapes remain visible in Standard View mode but are hidden in Wide View mode.

If a dashboard is in Wide View mode, and Available Shapes are moved to Shapes Hidden in Wide Mode, and back again, those shapes remain hidden. To make those shapes visible, the dashboard must be set to Standard View mode, before shapes are moved back to Available Shapes.

5 **Optional: To reload the original settings, click Restore.**

6 **Optional: To clear Available Shapes and Shapes Hidden in Wide Mode, click Clear.**

7 **Click Apply.**

8 **Optional: To open the dashboard in Wide View mode, select Start in Wide Mod,, and click Apply.**

Wide View and Standard View Mode Controls

Wide View Properties includes a button on the dashboard top panel that toggles between Wide View and Standard View modes. When the button is clicked, the view for all configured frames in the document changes.

Note:

Clicking the button shows or hides specified shapes. If Wide View charts and pivots are configured, clicking the button, enables users to toggle between Standard and Wide View charts and pivots.

Caution!

If Wide View objects are configured, it is recommended that ↔ and ▤ are copied to at least one other frame for convenience. As configuration frames are discarded from runtime documents, you must ensure that ↔ and ▤ are available from at least one other frame, so you can toggle between Wide View and Standard View mode in a runtime document.

➤ To copy ↔ and ▤ to other frames:

1 **In Interactive Reporting Studio, select View, then Unhide Section.**

2 **Select** `Qiq_SwitchChartPivot`**, and click OK.**

3 **Press Ctrl+D to enter Design mode.**

4 **Select all four images in the top right corner, by pressing the Ctrl key.**

Begin with the transparent box and work along to the gray image. Copy the images in the specified order.

5 **Press Ctrl+C to copy the images.**

6 **Navigate to a frame, and press Ctrl+V to paste the images.**

7 **With all four images selected, on the section toolbar, click Left and click Top to justify the images.**

8 **Double-click the top image, named *picHideShow^Qiq_SwitchChartPivot,* select Border and Background, and set the border to None.**

9 **Click OK.**

10 **Press Ctrl+D to exit Design mode.**

11 **Click the button to test switching between Wide View and Standard View mode.**

See “Creating Frame Prototypes” on page 235, for information on the Wide View feature when creating a frame prototype.

Using the Pivot Fact Properties Frame

Pivot Fact Properties enables you to specify fact columns that are hidden when a dashboard is initially opened. Hidden columns can be made visible by the end user if required.

➤ To configure pivot facts:

1 **In Interactive Reporting Studio, navigate to Pivot Fact Properties.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **From Select Dashboard Section, select a section.**

A list of available pivots are displayed.

3 **Optional: To view the section, click 🗔.**

4 **Select a pivot.**

5 **Optional: To view the pivot data, click 🗔.**

6 **Move selected fact columns between Facts Hidden Initially and Facts Visible Initially by clicking > and <.**

7 **Configure other sections.**

8 **Click Apply.**

Using the QIQ Sort Properties Frame

When the Sort feature is first merged into an Interactive Reporting document, no sort specifications are active, specifications must be configured with QIQ Sort Properties. The Sort feature provides default sorting and configured sorting options. These options are set in QIQ Sort Properties. Default sorting is recommended based on simplicity and general applicability. With default sorting the developer need only specify whether sorting is available on a frame or not. For example, you can select to sort charts and pivots by columns that are present in the charts or pivots on that frame.

➤ To configure sort specifications:

1 **In Interactive Reporting Studio, navigate to QIQ Sort Properties.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **Select an option.**

- All Dashboards—All frames are available
- Configured Dashboards—Only frames with applied sort configurations are available
- Unconfigured Dashboards—Only frames without applied sort configurations are available

3 **From the drop-down list, directly under the options, select a frame.**

Available frames vary depending upon the selected option.

4 **Optional: To view the selected section, click ▦.**

To make all rows or x-axes available for sorting, select Use any existing Side Labels / X Categories.

➤ To configure individual sort columns:

1 **In Manual Sort Configuration, from Embedded Sections, select a section.**

2 **Optional: To view the selected section, click ▦ (next to Embedded Sections).**

Available Columns are displayed.

The selected option, Row / X or Column / Depth, determines the Available Columns that are displayed.

3 **From Sort Options, select a sort function.**

4 **Move selected columns between Available Columns and Sorted Columns by clicking › and ‹.**

5 **Repeat steps 1–4 to configure all sort specifications.**

6 **Optional: To return to a previously applied sort configuration, click Restore.**

7 **Optional: To remove all applied sort configurations, click Clear.**

8 **Click Apply to assign the specifications.**

Configured columns are displayed in the drop-down list when ↑ or ↓ is clicked on the frame.

9 **Save the dashboard.**

If ↑ and ↓ are added to all frames, they are only visible in frames with applied sort specifications.

Embedded sections must be associated in the Associate step of Dashboard Studio to enable them to be manipulated by the Sort feature. See "Step 4" on page 45.

After sort is applied, references to objects are stored as a Quick Sort property in the Interactive Reporting document. If these referenced objects (charts, pivots, tables, and columns) are removed or renamed, the configuration is obsolete, as these objects can no longer be found. The Sort feature verifies the configuration as it is loaded, and ignores obsolete elements. Loading takes place when QIQ Sort Properties is first activated, and when Apply is clicked.

A change to a configured section; for example, section names changed or removed, implies QIQ Sort Properties must be activated. In addition, Restore and Apply must be clicked, so the specification can be reloaded, corrected, and made permanent.

Using the List Expansion Properties Frame

The List Expansion feature enables you to see a full description of a selected item in a list or a drop-down list. Items displayed in lists may not be logically truncated or abbreviated as acronyms, so this feature enables you to extend the floating comments function to view a description as an extended text label.

List Expansion Properties is enabled only if lists or drop-down lists are available on the frame. The selected list or drop-down list must include the standard control code:

```
ActiveDocument.Sections[txlMe.Text].Qiq_onControlClick(this.Parent,this)
```

Note:

Quick Filters lists in dashboards created with a pre-9.0 Release template exclude this line of standard control code. The code must be manually added to Quick Filters lists to apply list expansion.

➤ To use List Expansion Properties to expand items:

1. **In Interactive Reporting Studio, navigate to List Expansion Properties.**

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. **From List Item Expansion, select Include Dropdown lists to add drop-down lists to Available Lists.**

3. **Move selected lists between Available Lists and Lists to expand by clicking > and <.**

4. **Click Apply to assign the lists to be expanded.**

5. **Navigate to a frame that contains a list, and test the feature by selecting an item to be expanded.**

Note:

List items are only displayed in the floating comment if the line of code is added to the selected list.

➤ To add the standard control code to a list or drop-down list:

1. **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2. **Select the list or drop-down list, right-click, and select Scripts.**

   Scripts Editor is displayed.

3. **In Scripts Editor, perform an action:**

- Enter this line of code

  ```
  ActiveDocument.Sections[txlMe.Text].Qiq_onControlClick
  (this.Parent,this)
  ```

  (In some template versions, `ActiveSection` replaces `this.Parent`. This does not affect the execution of the code)

- Copy and paste the line of code from another control (for example, a button) on the frame to avoid data entry errors

4. **Click OK.**
5. **Repeat steps 1-4 for all other lists or drop-down lists to be expanded.**
6. **After the list is configured, press Ctrl+D to exit Design mode.**

Using the Language Properties Frame

Dashboard Studio templates contain strings for all supported languages. These include English, French, German, Spanish, Dutch, Portuguese (European), Portuguese (Brazilian), Japanese, Simplified Chinese, Traditional Chinese, and Korean. As a result, the dashboard may be slightly larger.

The Language Properties frame enables users to remove unnecessary languages from templates, editable source master (ESM) documents, or runtime dashboards. The frame is available in Release 8.5 and later templates.

➤ To use the Language Properties frame to remove languages:

1. **In Interactive Reporting Studio, navigate to Language Properties.**

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. **Move selected languages between Available Languages and Languages to Remove by clicking > and <.**

3. **Perform an action:**

   a. Apply changes immediately.

   - Click Apply Now to remove languages from the current document (The operation is not reversible. A dialog box is displayed to confirm that the selected languages are to be removed)
   - Click OK (The selected languages are removed from the ESM)

   b. Click **Apply at Runtime** to remove languages when the document is converted into a runtime release.

   In this case, only the runtime document is affected and all languages are still available in the template or ESM.

Note:

English is the default language for dashboards and is not available for deletion.

Using the Date Formats Frame

Custom date formats for the As of Date can be entered and maintained in Interactive Reporting documents using the Date Formats frame. The date formats are available when defining the As of Date in "Step 7: Configure Properties" on page 75 of Dashboard Studio.

➤ To use the Date Formats frame:

1 **In Interactive Reporting Studio, navigate to Date Formats.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **In Manage Custom Date Formats, enter a date format.**

For example, enter `yyyy-mm-dd`.

3 **Click Test to validate the format.**

A text label displays the formatted date.

4 **Click Add to append the format to the Defined Date Formats.**

5 **Selecting a date format in Defined Date Formats enables these controls.**

a. **Optional:** To delete a Defined Date Format, select a format, and click 🗑.

b. **Optional:** Reorder Defined Date Formats as they are displayed in Step 7 of Dashboard Studio, by selecting a format and clicking ▲ or ▼.

6 **Perform an action:**

- To reload the original settings, click Restore
- To remove the current settings, click Clear
- To load the date format, click Apply

Using the Color Palette Properties Frame

Developing a color palette creates a set of palette definitions within Interactive Reporting documents. These definitions are available to be used by components, such as Chart Shade (see "Configuring the Chart Shade Component" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*) or the Chart Spotlighter feature (see "Chart Spotlighter Feature" on page 40 or "Configuring for Chart Spotlighting" on page 197).

➤ To use the Color Palette Properties frame:

1. In Interactive Reporting Studio, navigate to **Color Palette Properties.**

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. In **Load External Palette File**, enter the full path for an externally defined palette file.
3. **Optional:** Click **Browse** to locate a palette.
4. Click **Add** to append the palette to Available Palettes.

   Available Palettes displays the palettes that are available in the Interactive Reporting document. If a palette with an identical name exists, a number suffix is added to the file name to make it unique.

5. From **Available Palettes**, select a palette, and click to duplicate the palette.

   A copy is created by appending a number suffix to the name. For example, a palette called *Sporty* is copied as Sporty1. is selectable when an item is selected in the list.

6. **Optional:** To remove a palette, select an available palette and click .
7. In **Selected Palette Name**, enter a replacement name, and click to rename the palette.
8. In **Click to Change Color**, select a colored bar to change individual palette colors.

   The color picker dialog box is displayed.

9. Select a color and click **OK**.

   The bar color is changed.

10. Perform an action:

    - To save the palette to the path entered in step 2, click Export (The path can be a file name or a folder with a trailing backslash (\). Export is not selectable if Load External Palette File is blank)
    - To reload the original settings, click Restore
    - To remove all current entries, click Clear All
    - To load the color palette properties, click Apply

Using the Webdash Template to Build Dashboards

You can use `Webdash_Template.bqy` to build dashboards in environments (such as pure HTML, Web plug-ins, or desktop applications) that process Interactive Reporting documents.

The build tool is a Dashboard Studio configuration frame, called *Build Dashboard,* that uses the placement features of the object model. These features enable you to create, place, and size dashboard objects without going into Design mode.

Webdash eliminates the need for the Dashboard Studio wizard in most cases. The eight steps have become one configuration frame. However, the wizard is required to combine documents and templates (using the Dashboard Studio Merge Utility), create and customize templates, configure styles, and build more sophisticated, user-configured dashboards.

Webdash is available in multiple formats and can be used in multiple ways:

- `Webdashbuilder_sample_esm.bqy`—Located in the docs\samples directory, and provided as an example Webdash document.
- `Webdash_Template.bqy`—Located in the \DDS\templates directory. The template provides a starting point for a Webdash document. You can launch the template from the Dashboard Studio wizard to create a blank ESM, and then create data sections to be added later to dashboard panels. Alternatively, the Import feature can be used to import existing data sections. The template can also be customized by adding additional controls; for example, Quick Filter objects, or apply a style sheet by connecting the template to the Dashboard Studio wizard.
- `Webdashbuilder_component.bqy`—Located in the DDS\components\WebDashBuilder directory. It contains the code, configuration frame, and prototype sections. The component can be merged into existing documents so that non-Webdash templates or ESMs can leverage the Webdash functionality. See "Webdash Builder Component" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

Frame prototypes are available in the `Webdash_Template.bqy` and `Frame_Library.bqy`. These files are located in the templates folder.

Tasks that you perform to use the Webdash template to create dashboards:

1. Launch Dashboard Studio wizard, and perform an action:
    - Double-click the Webdash template to create a blank ESM, and add your data sections
    - Select the Webdash template, and click ⇥ to import existing data sections
2. **Optional:** Customize the template and include in the Dashboard Studio wizard.
3. Create local filters.
4. Use the Build Dashboard configuration frame to create dashboard sections.

Creating Local Filters

Using Build Dashboard to create local filters is equivalent to setting filters in the QIQ Filters Properties frame, (see "Using the QIQ Filter Properties Frame" on page 83).

Note:

The Local Filters section is only available on QIQ Filter Properties if Webdash is *not* available in the Interactive Reporting document. Where Webdash is available, create the local filters using the Build Dashboard frame.

➤ To use the Build Dashboard frame to create local filters:

1. **In Interactive Reporting Studio, open the dashboard created from the Webdash template.**
2. **In Create Local Filters, select a results section.**

   The results columns are displayed.
3. **Select Columns.**
4. **Move selected columns between Columns and Local Filters by clicking › and ‹.**
5. **Click Apply at this point in development, or build your dashboards before clicking Apply.**
6. **Navigate to another dashboard frame, and verify that the local filters function correctly.**
7. **From Filters on the top panel or Quick Filters, select and set a filter.**

   The local filter selected in Build Dashboard is displayed in the Filters or Quick Filters, and the dashboard data changes to reflect the selected filter.
8. **Save the dashboard.**

Tip:

To reorder local filters, use the Filter Aliases Properties Frame. See "Filter Aliases Properties Frame" on page 86.

Building Dashboards

This topic expects that charts and queries have been built, either created from scratch or imported using Dashboard Studio Merge Utility.

Note:

You can use the Build Dashboard frame in Interactive Reporting Studio or EPM Workspace to build dashboards.

➤ To use the Build Dashboard frame to create dashboards:

1. **In Interactive Reporting Studio or EPM Workspace, open the dashboard from the Creating Local Filters procedure.**
2. **From Sections, select Build Dashboard.**
3. **In Create and Configure Dashboards, enter a dashboard name.**

   For example, enter `Summary`.
4. **From Prototype, select a layout, and click (Add).**

   For example, select 4 (2 x 2) Round |–|–|. The code reads the geometry of the panels (for example, 4 round panels with 2 panels at top and 2 panels at the bottom) and creates a miniature facsimile of the layout.

The dashboard name is displayed in Navigation Targets and above the preview frame.

5. **Optional:** To change the name or panel layout, enter another **Dashboard Name** or select another prototype, and click ↻ (Modify).

Caution!

If you rename a dashboard frame to *Item*, an error is displayed in the Interactive Reporting Studio Console window.

6. From **Available Objects**, select charts, pivots, or tables, and click the preview frame to display the icons.

   Chart and pivot icons are displayed on the preview frame, in a left-to-right, top-to-bottom order. Charts and pivots that share a name prefix are displayed on the same panel. For example, Revenue by Item Type Chart and Revenue by Item Type Pivot are displayed on the same panel.

7. **Optional:** Select multiple available objects, and click > for the best-guess option.

   > is selectable only if the preview frame is empty. When an object is added to a panel, > is not selectable.

8. Repeat steps 3–7 to build another frame.

9. **Optional:** Reorder **Navigation Targets** by selecting a target and clicking ˄ or ˅.

   After Apply is clicked, this reorders the navigation of frames in the dashboard.

10. Click **Apply** to create the dashboard.

Design Notes

- Chart, pivot, and table icons that are added to the preview frame are displayed in the corners of the panel. However, in the dashboard that is created, the charts, pivots, and tables fill the panel.
- Selecting an available object and clicking a panel results in the object being displayed on the panel. If an icon is clicked in a panel, it is removed from view in the panel. Clicking an icon in one panel, and then clicking in another panel adds the removed icon to the second panel.
- Only one chart icon and one pivot or table icon can be displayed in a panel. If a chart icon, and a pivot or table icon are added to a 1 Round or 1 Square panel layout, they occupy the same panel, and the switch between a chart and pivot view feature is automatically applied. See "Switching Between Chart and Pivot Views" on page 34.
- The names of navigation targets and prototypes can be modified before Apply is clicked, by clicking ↻.
- Changing the layout prototype may affect the displayed data. For example, moving from 4 panels to 2 panels results in a loss of displayed data, and moving from 1 panel to 2 panels results in additional data requiring configuration.

Embedding Sections

Webdash recognizes only embedded section shapes with names in the form Chart1, Chart2, or Pivot1, Pivot2, and so on. The name *Chart* applies to any graphical view, and the name *Pivot* applies to any tabular view. The number indicates the panel on which the object is placed.

If a dashboard is built by using drag-and-drop in Interactive Reporting Studio, then the names generated by Interactive Reporting are incorrect for results, tables, and reports, and Webdash cannot recognize them. To edit these dashboards in Webdash, the generated names must be modified to conform to names that Webdash expects.

Embedding Browsers

Embedding browsers relies upon script-based object creation and placement features. See "Object Model Placement and Sizing" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 1: Dashboard Design Guide.*

➤ To use the Build Dashboard frame to create dashboards that contain embedded browsers:

1. In Interactive Reporting Studio or EPM Workspace, open a dashboard that contains the Webdash_builder sample.
2. From **Sections**, select **Build Dashboard**.
3. **Optional:** Use the **Create Local Filters** section to set local filters.
4. In **Create and Configure Dashboards**, enter a dashboard name.

   For example, enter `Web Browser`.

5. From **Prototype**, select a layout, and click (Add).

   For example, select 1 Round.

   The dashboard name is displayed in Navigation Targets and above the preview frame.

6. **Optional:** To change the name or panel layout, enter another **Dashboard Name** or select another prototype, and click (Modify).

Caution!

If you rename a dashboard frame to *Item*, an error is displayed in the Interactive Reporting Studio Console window.

7. From **Available Objects**, select **Set Embedded Browser**, and enter a URL.

   For example, enter www.hyperion.com, for the Oracle Web site.

8. Click the preview frame to display the browser icon.
9. Repeat steps 3–8 to build another frame that contains an embedded browser.
10. **Optional:** Reorder **Navigation Targets** by selecting a target and clicking ▲ or ▼.

After Apply is clicked, this reorders the navigation of frames in the dashboard.

11 **Click Apply to create the dashboard.**

Embedding Reports

Embedding reports relies upon script-based report object creation and placement features. See “Embedding Reports” on page 228.

➤ To use the Build Dashboard frame to create dashboards that contain embedded reports:

1 **In Interactive Reporting Studio or EPM Workspace, open a dashboard that contains the Webdash_builder sample.**

2 **From Sections, select Build Dashboard.**

3 **Optional: Use the Create Local Filters section to set local filters.**

4 **In Create and Configure Dashboards, enter a dashboard name.**

For example, enter `Summary Report`.

5 **From Prototype, select a layout, and click ⧉ (Add).**

For example, select 1 Square.

The dashboard name is displayed in Navigation Targets and above the preview frame.

6 **Optional: To change the name or panel layout, enter another Dashboard Name or select another prototype, and click ↻ (Modify).**

Caution!

If you rename a dashboard frame to *Item*, an error is displayed in the Interactive Reporting Studio Console window.

7 **From Available Objects, select a report, and click the preview frame to display the icon.**

8 **Optional: Select multiple available objects, and click > for the best-guess option.**

> is selectable only if the preview frame is empty. When an object is added to a panel, > is not selectable.

9 **Repeat steps 3–8 to build another frame that contains an embedded report.**

10 **Optional: Reorder Navigation Targets by selecting a target and clicking ∧ or ∨.**

After Apply is clicked, this reorders the navigation of frames in the dashboard.

11 **Click Apply to create the dashboard.**

Note:

When embedding a report into a Webdash document using the Build Dashboard frame, the name of the report must begin with Pivot to be recognized as a tabular structure.

Verifying Dashboard Functionality

➤ To test dashboard functionality:

1. **In Interactive Reporting Studio, navigate to the dashboard sections created in the Building Dashboards procedure.**
2. **Use the frame navigation to move between the sections.**

   For example, if a tab prototype was created, use tabs.
3. **Optional: If a chart and a pivot or table is configured on a 1 Round or 1 Square panel frame, click 📊 or ▦, to switch between the views.**
4. **Use other dashboard functions, by selecting controls on the top panel.**

   For example, click ↑ or ↓, to sort data.
5. **Optional: Set filters using Quick Filters.**
6. **Save the dashboard.**

Using Dashboard Studio to Customize Dashboards

A dashboard that is created using Build Dashboard can be customized (but does not have to be) in Dashboard Studio. The dashboard properties and styles can be configured by using "Step 7: Configure Properties" on page 75 and "Step 8: Configure Style" on page 79. The dashboard can be merged or imported into another document by using the "Import Feature" on page 117.

Layout Manager

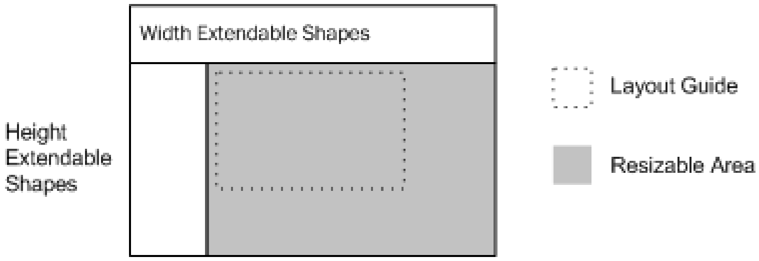
Note:

Layout Manager requires Interactive Reporting Studio version 11.1.1 or later. If a dashboard with the Layout Manager feature is modified in a pre-11.1.1 environment (new frames are added or existing frames modified using the WebDash feature), the modified frame takes the original size of the frame prototype from which it was derived. As a result, the modified frame may be a different size than the rest of the frames in the dashboard.

Layout Manager adapts dashboard layouts to various sizes of screen real estate. It does this by resizing content to fit the available screen area when a dashboard section is activated. This happens when you move to a section or apply a filter. When the screen area changes as a result of hiding or revealing the catalog, or when the size of the window changes, no event is fired, and you must click to trigger the resize operation.

Figure 6 shows the basic dashboard layout.

Figure 6 Top Left Fixed Layout



The unshaded areas typically contain controls, such as button bars, option buttons, list boxes, navigation tabs, and corporate logos. The shaded areas typically contain content, such as pivots, tables, charts, and reports.

When you activate a section or click , the content shrinks or expands. The unshaded control areas expand although no changes are made to the controls within the borders. That is, a side panel background can grow downwards (height-extendable), or a top panel background can grow sidewards (width-extendable).

The developer defines the shapes or objects that reside in the resizable area by positioning a rectangle, known as the layout guide, to the frame.

Shapes that have a top left corner inside, below, and to the right of the layout guide are included in the resizable area (shaded area in Figure 6). When you expand the application window, the shapes expand to fill the space. The layout guide also marks the minimum size to which the resizable area can shrink.

Typically, controls in the side and top panel are fixed; however, Layout Manager shrinks or expands the Top Panel and Side Panel backgrounds along with the content area so the dashboard retains a regular rectangular shape.

- Top Panel Background—Fills the horizontal space. This shape is part of the Width Extendable Shapes collection.
- Side Panel Background—Fills the vertical space. This shape is part of the Height Extendable Shapes collection.

➤ To restore a screen to the dashboard's original design, click . The shapes restore to the positions and sizes in which they were originally saved.

Resizing Dashboard Section Content

➤ To resize the content of a dashboard section:

1 **In Interactive Reporting Studio or through EPM Workspace, open a dashboard.**

The activated screen is resized to fit the available area.

2 **Change the size of the available screen area by dragging the bottom right corner of the window, or by changing the size of the window's catalog area.**

3 **Click to resize the content.**

The shapes expand or shrink according to the available screen real estate.

Note:

If you shrink the screen area so that the layout guide no longer fits, the content is only resized as far down as the size of the layout guide.

Showing and Hiding Layout Guides

On user frames, the layout guide is hidden by default.

- To show layout guides, go to Layout Properties and select Show Layout Guides. (Red layout guides are visible in dashboard sections when in Design mode.)
- To hide the layout guides again, deselect Show Layout Guides.

Layout Manager and Dashboard Studio Wizard

Layout Manager operates on frames that exist when a dashboard's startup scripts execute. Frame prototypes are never resized. Dashboard Studio Wizard calls the startup scripts when settings in Steps 4 - 8 of the wizard are applied. As a result, new frames are under the control of Layout Manager.

If a dashboard is not in Design mode, then content for a section is resized when that section is activated based on the default layout guide associated with each frame. The layout guide is hidden by default, so it does not interfere with dashboard objects.

When a Dashboard is in Design mode (for example during Step 3 - Layout in the Wizard), no Activation events are fired, and no resizing occurs.

When dashboard object creation and placement is complete, move to Step 4 - Associate in the wizard. When you click , Layout Manager:

- Examines all dashboard frames and classifies objects as Resizable, Height Extendable, and Width Extendable based primarily on the position of the top left corner of the layout guide
- Records the sizes and positions of the new objects so that they can be restored at any time by clicking .

Objects whose default layouts are already saved are not updated. To update the default layout of every resizable shape on a section, move to the Layout Properties frame and follow the steps described in "Saving Dashboard Designs as Default" on page 109."

Using the Layout Properties Frame

Use the Layout Properties frame to:

- Generate layout objects for new dashboard sections
- Save Dashboard Designs as the default
- Customize the resize behavior of individual sections by configuring Layout Properties
- Change the minimum size of a dashboard section
- Disable Layout Manager

Generating Layout Objects

Layout objects that enable resize behavior are automatically generated for existing dashboard sections that do not have layout objects when a dashboard starts up. However, if you are creating a new dashboard section, click Apply in Step 4 of the Dashboard Studio Wizard to generate layout objects and enable dashboard resize behavior.

Note:

You cannot resize dashboard sections that did not go through the Associate wizard step or that were not created using the Build Dashboard form in a WebDash document. If your dashboard is not connected to the Wizard, however, you can still generate layout objects manually.

➤ To generate layout objects:

1 **In Interactive Reporting Studio, open a dashboard and navigate to Layout Properties.**

To access the configuration frame, use Sections or the Configuration drop-down list in Dashboard Studio.

2 **With one or more sections selected, click to:**

- Create the layout guide and position it using best-guess options
- Display Available Shapes
- Populate Resizable Shapes, Height Extendable Shapes, and Width Extendable Shapes based on the location of the layout guide

- Add Resize and Restore buttons to the section, if at least one button does not exist in the section

  By default, these buttons are located to the right of the Dashboard Studio toolbar. You can move these buttons if desired.

Note:

As seen in Figure 6, the layout guide defines the top left corner of resizable shapes and the minimum area of a dashboard section. If you are not satisfied with the default location of the layout guide, see “Repositioning Layout Guides” on page 109. If you are not satisfied with the default size of the layout guide, see “Changing the Minimum Size of Dashboard Sections” on page 111".

Repositioning Layout Guides

Layout guides are generated using best-guess options. If a guess is wrong and the layout guide is not positioned in the desired location, undesirable resizing behavior may occur.

➤ To reposition a layout guide:

1 **Select Show Layout Guides.**

2 **In the dashboard section, reposition the layout guide as desired.**

3 **Navigate to Layout Properties, select the dashboard section, and click .**

Shape collections are re-populated based on the new location of the layout guide.

4 **Optional. Deselect Show Layout Guides to hide the layout guides.**

Saving Dashboard Designs as Default

When you finish designing one or more dashboard sections, save the current dashboard designs before they are resized by Layout Manager. This way, when a user clicks in a dashboard section, Layout Manager restores the section to your saved design.

➤ To save a dashboard design as default:

1 **In Interactive Reporting Studio, open a dashboard and navigate to Layout Properties.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **From Dashboard Sections, select one or more dashboard sections.**

3 **Click to move to the selected dashboard.**

The dashboard is activated and resized.

4. **Optional.** Go into Design mode, fine tune any shape placements and sizes, and exit Design mode.
5. Use the back button to return to Layout Properties.
6. Click to save the current state of the selected dashboard section as its default design.
7. Repeat steps 2 - 6 for each section

Note:

Dashboard designs are automatically saved when you create a dashboard section in the Wizard and associate it in Step 4, or when you use Layout Manager to upgrade a pre-Release 11.1.1 dashboard to the current release.

Note:

Designers use ESM dashboards, whereas end users most likely use runtime dashboards. The Section Catalog pane is visible in ESM, whereas it is hidden in runtime. This means that the available screen area for dashboard content is greater for end users than it is for designers. Designers should compensate for this by using a higher resolution display to get the extra screen area required to see both the catalog and the end users larger content area, or by using the same resolution display as the end users and hiding the catalog.

Configuring Layout Properties

Best-guess configurations do not always produce the right resize behavior in dashboard sections. By setting a shape to be resizable or not resizable, a dashboard designer can customize the resize behavior of a particular section.

➤ To configure layout properties:

1. In Interactive Reporting Studio, open a dashboard and navigate to Layout Properties.

   To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2. From **Dashboard Sections**, select a dashboard section.
3. Move selected **Available Shapes** between **Resizable Shapes, Height Extendable Shapes,** or **Width Extendable Shapes** by clicking > or <.
   - Shapes in **Available Shapes** are not resized.
   - **Resizable Shapes** resize to fit within the available screen real estate.
   - **Height Extendable Shapes** only resize vertically, and typically contain a dashboard section's side panel shape.
   - **Width Extendable Shapes** only resize horizontally, and typically contain a dashboard section's top panel shape.
4. Perform an action:

- To discard in-memory settings and revert to original shape assignments, click **Restore**.
- To remove all shape assignments, click **Clear All**.
- To remove shape assignments from selected frames, click **Clear**.
- To commit shape assignments, click **Apply**.

5 **Click to navigate to a customized section, and click to restore the section to its original default design.**

6 **Check to see if any shapes are 'stuck.'**

Shapes become 'stuck' and do not restore along with the rest of the section because the shapes are new and have not yet been given a default design. If one or more shapes are 'stuck':

a. Go into Design mode.

b. Move and resize the 'stuck' shapes so they are proportional to the rest of the restored section.

c. Exit Design mode.

d. Navigate to Layout Properties, and click to save the design of the dashboard.

7 **Repeat Step 5 and Step 6 for all customized dashboard sections.**

Changing the Minimum Size of Dashboard Sections

You can change the minimum Resizable Area by manually showing the layout guide and resizing it. However, you may wish to use the size of the current application window as a point of reference.

➤ To change the minimum size of dashboard section:

1 **Go to a dashboard section and shrink the application window to the desired minimum size.**

2 **Navigate to Layout Properties.**

3 **Select a dashboard section and click .**

The minimum size of the selected sections is set to the current application window size.

4 **Click Apply.**

Disabling Layout Manager

You can use Layout Properties to disable one or more dashboard areas if desired.

➤ To disable resize behavior for a dashboard section:

1 **Navigate to Layout Properties, and select one or more dashboard sections.**

2 **Click Clear.**

3 **Click Apply.**

➤ To disable resize behavior for the entire dashboard

1 **Navigate to Layout Properties, and select Disable Layout Manager.**

2 **Click Yes in the dialog box that appears to revert dashboard sections to their default designs.**

3 **Click Apply.**

Updating a Pre-11.1.1 Release Dashboard to Include Layout Manager

Release 11.1.1 comes with two types of update kits. Using `<Javascript_Update_ResizeOn_dds>` adds the Layout Manager component and configures the resize properties and default settings.

➤ To update a pre-11.1.1 Release dashboard to be resizable:

1 **In the Dashboard Development Services Update Utility, update the Interactive Reporting document using the Layout Manager script, `JavascriptUpdateResizeOn_dds.js`.**

The document is marked as auto-resize.

2 **Open the Interactive Reporting document.**

Shapes and controls on the dashboard are automatically resized. On each dashboard section, Layout Manager buttons are added to the right of the Dashboard Studio toolbar. Move the buttons to more appropriate locations if desired.

3 **Test the resize behavior of each dashboard section by moving to the section, and clicking ↙ and ↗.**

If there are problems with resize behavior, see "Troubleshooting Unexpected Resize Behavior" on page 112.

4 **Click Apply and save the document.**

Tip:

If you are updating a pre-11.1.1 Release Interactive Reporting document that is not resizable, use the default script `JavaScriptUpdateConfig_dds.js` during the update process.

Troubleshooting Unexpected Resize Behavior

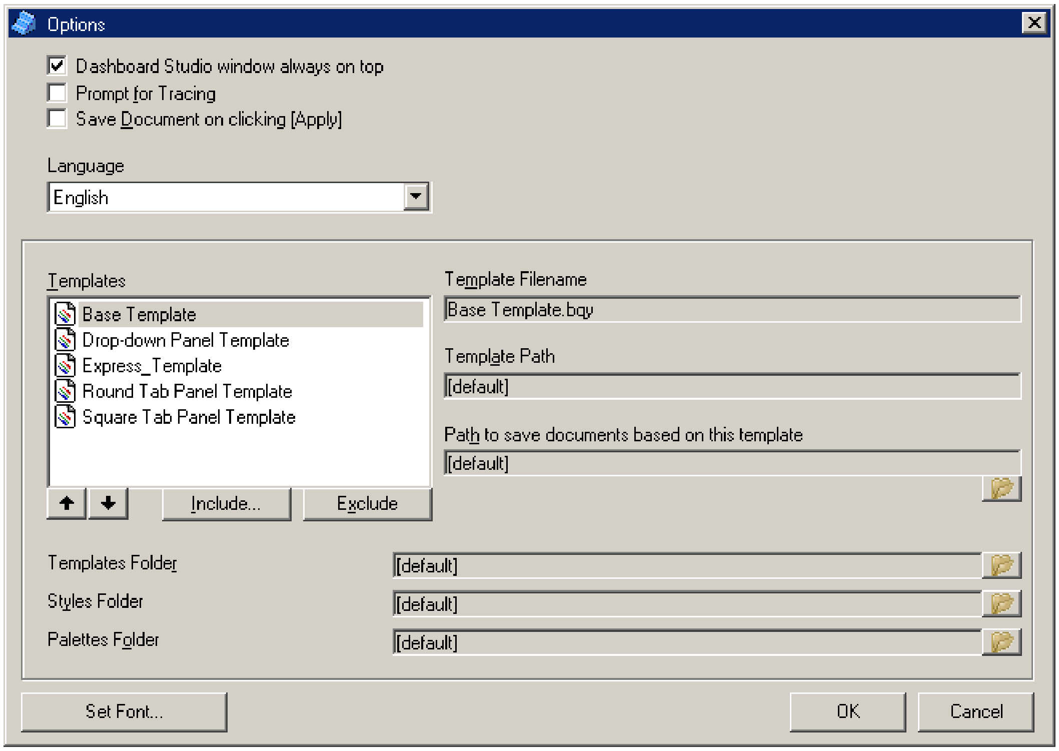
Layout Manager resizes dashboard sections using best-guess options. It works best with dashboard designs similar to those supplied with Dashboard Studio. If you experience unexpected resize behavior, see Table 8 for troubleshooting tips.

Table 8 Troubleshooting Unexpected Resize Behavior

Problem	Solution
Shapes in the resizable area do not shrink and grow as the screen area changes.	The layout guide may be lower or to the right of the top left corner of the shapes. Either reposition the Layout Guide, or move the list of shapes from the Available list to the Resizable Shapes list in the Layout Properties frame. See “Repositioning Layout Guides” on page 109 and “Configuring Layout Properties” on page 110.
The minimum size of a dashboard content makes it difficult to see.	Increase the size of the layout guide by dragging the bottom right hand corner outwards. See “Changing the Minimum Size of Dashboard Sections” on page 111".
Clicking restores a dashboard section to an inappropriate default layout.	Redesign and save a new default layout. See “Saving Dashboard Designs as Default” on page 109".
You want to disable resize behavior for some but not all dashboard sections.	Select the dashboards you do not want to resize and use the clear button on the Layout Properties frame. See “Disabling Layout Manager” on page 111".
You want to disable resize behavior for the whole document.	Use the Disable Layout Manager check box. See “Disabling Layout Manager” on page 111".
You want to have one or more shapes grow in width (or height) only as the window size changes.	Remove the shape from it current list in the Layout Properties frame and add it to the list of Width Extendable Shapes (or Height Extendable Shapes). See “Configuring Layout Properties” on page 110.

Options Window

Click on Select a Framework Template to open the Options window, and change the settings and environment of Dashboard Studio.



☑ Dashboard Studio window always on top
☐ Prompt for Tracing
☐ Save Document on clicking [Apply]

Language: English

Templates:
- Base Template
- Drop-down Panel Template
- Express_Template
- Round Tab Panel Template
- Square Tab Panel Template

Template Filename: Base Template.bqy

Template Path: [default]

Path to save documents based on this template: [default]

Templates Folder: [default]

Styles Folder: [default]

Palettes Folder: [default]

These elements are available in Options:

- **Dashboard Studio window always on top**—Determines whether Dashboard Studio is on top of other application windows, and is selected by default
- **Prompt for Tracing**—Enables Dashboard Studio and the dashboard to record JavaScript tracing information in the Console window and the Diagnostics frame (The feature should be set only if you are trying to resolve a problem with Dashboard Studio or a dashboard)
- **Save Document on Clicking** [**Apply**]—Causes Interactive Reporting Studio to save the dashboard every time selections are applied
- **Language**—Enables the language option of the Dashboard Studio wizard-interface to be set (A necessary option for localized or Unicode releases, as different languages display characters differently)
- **Templates**—Contains a list of aliases that represent the configured templates available to users of Dashboard Studio (Add templates to the list by clicking Include and selecting a template from the Templates folder. Remove a template from the list by selecting the template, and clicking Exclude. Reorder the displayed templates in Select a Framework Template by selecting a template from Templates, and clicking ↑ or ↓)

  Note:

  A template alias can be renamed. By default, a template uses the file name that it represents. A template may be included multiple times with multiple save paths.
- **Template Filename**—Displays the file name of the template associated with the template alias

Note:

Default paths (folders such as templates, documents, styles, palettes) reside in the Dashboard Studio location.

- **Template Path**—Displays the path location of the configured templates
- **Path to save documents based on this template**—Displays the path where dashboards derived from the template are saved (To change the path, click 📁, and navigate to another location. A template may be included multiple times with multiple save paths)
- **Templates Folder**—Displays the path location of the folder of configured templates (To change the path, click 📁, and navigate to another folder)
- **Styles Folder**—Displays the path location of the Styles folder (Click 📁 to change the path)
- **Palettes Folder**—Displays the path location of the Palettes folder (Click 📁 to change the path)
- **Set Font**—Launches a Font dialog box to enable the font style to be configured for the Dashboard Studio wizard-interface (A necessary option for localized or Unicode releases, as different fonts display characters differently. Users must select an appropriately sized font that suits the UI, otherwise characters may be too large and cut off)

5

Using the Import Feature

In This Chapter

Import Feature

The Import feature offers an invaluable set of tools to help automate the process of combining and updating various parts of one or more Interactive Reporting documents.

The Import feature contains Import and Merge tabs.

Import Tab

You use Import to import sections from Interactive Reporting documents into Dashboard Studio templates to create ESMs that Dashboard Studio can convert into dashboards. Thus, you reduce the time and effort required to create Interactive Reporting dashboards that contain queries, data models, charts, pivots, and reports.

Merge Tab

You use Merge to merge sections from two or more Interactive Reporting documents to create a composite document. Libraries of commonly used sections are developed, and reuse is encouraged among a range of implementations, saving time and promoting consistency. Merging also enables complex development tasks to be divided into smaller tasks so a team of developers can work in parallel.

Remove Duplicate Images from the Final Document

The Dashboard Studio Merge Utility and Import feature enables you to consolidate duplicate images into the Resource Manager. All instances of an image are changed to reference the single copy of the image in the Resource Manager. The Interactive Reporting document file size and memory footprint are reduced, which improves the loading speed, and makes reuse of existing images from the Resource Manager possible when creating dashboards and reports. See “Resource Manager” in the *Hyperion Interactive Reporting Studio User's Guide*.

Duplicate images are removed when Remove duplicate images from the final document is selected on the Import or Merge tabs.

After the import or merge process is complete, a report detailing the imported or merged sections is displayed. The number of removed duplicate images is recorded at the end of the report.

After the Import or Merge Process

After the import or merge process, sections may be renamed, because Interactive Reporting Studio does not permit a document to contain sections with identical names. However, renaming may cause the JavaScript that referenced the original name to break.

If the import or merge process fails, contact Oracle Customer Support with details of the failed document. In many cases, Oracle can explain the problem, so you can make changes and successfully import the document.

Accessing the Import Feature

To access the Import feature, from the Dashboard Studio Start window, click → or, in Create a Document From Template, right-click a template, and select Import.

Dashboard Studio Merge Utility is displayed.

Note:

If a Window Scripting dialog box is displayed, click No. You will not receive similar messages in the future.

Unicode Functionality

The Dashboard Studio Merge Utility and Import feature automatically convert old code page based Interactive Reporting documents to Unicode before use, enabling documents with different languages to be merged or imported. Only the resulting documents are converted, originals remain unchanged.

Using the Import Tab

Import, which makes assumptions about the documents being processed, can alert you to document problems and prevent the importation of sections that might render the resulting document unusable within Dashboard Studio.

Using the Import tab, you combine sections in three ways:

- Importing One to One
- Importing Many to One
- Importing Many to Many

Importing One to One

Import all non-Dashboard Studio sections from one Interactive Reporting document into a Dashboard Studio template to create one new ESM file.

➤ To import sections by using One to One:

1. **From Import Method, select One To One.**

2. **Optional: Select Select Sections.**

   Only sections from the Interactive Reporting document specified in Document to Import can be selected. All sections from the specified template are included in the imported document.

3. **Optional: Select Remove duplicate images from the final document.**

   See "Remove Duplicate Images from the Final Document" on page 117.

4. **Click 📂 (next to Template), to locate the template.**

   Select a Dashboard Studio template that provides the sections to be imported into the Interactive Reporting document to create an ESM. The Template box is pre-filled with the location of the template selected on the Start window.

5. **Click 📂 (next to Document to Import), to locate an Interactive Reporting document.**

   The original, selected Interactive Reporting document is not altered.

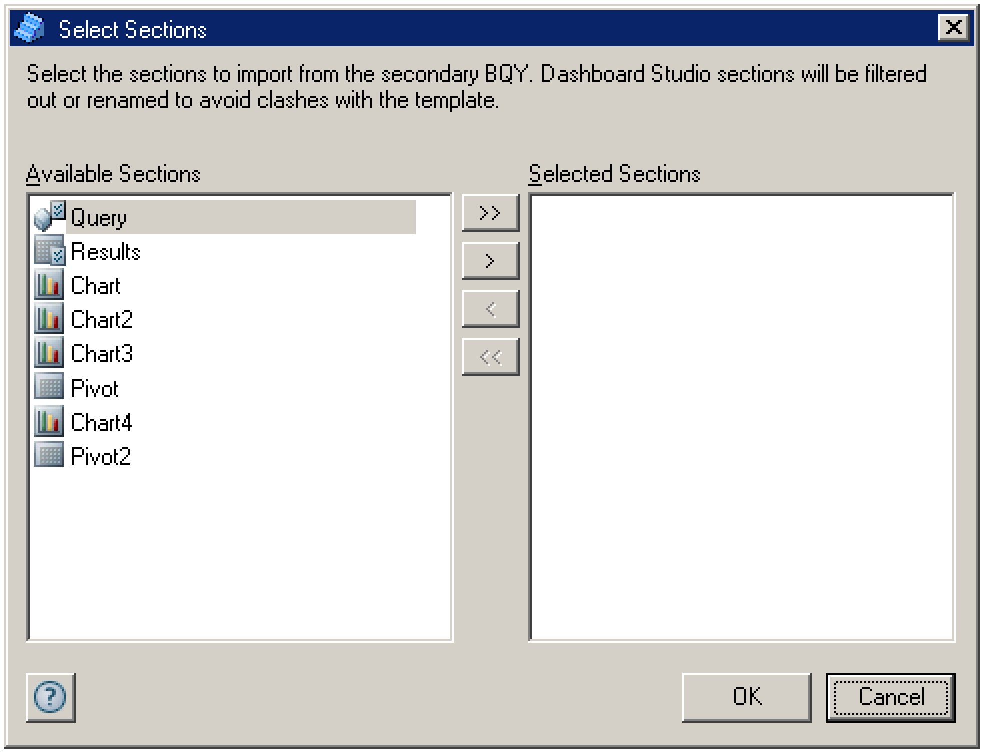
6. **Click 📂 (next to Save Path), to locate where the imported document is to be saved.**

   Import generates a save path that is based on the specified document to import. You can specify another name or save path.

7. **Click Import.**

   If sections are being imported, Select Sections is displayed.

   The document to import is inspected, and all document sections are displayed in Available Sections.



Select the sections to import from the secondary BQY. Dashboard Studio sections will be filtered out or renamed to avoid clashes with the template.

8 **Perform an action to move sections between Available Sections and Selected Sections:**

- Select one or more sections, and click > or <
- Click >> or << to move all sections
- Double-click a section

When a section is selected, all dependent sections are selected. For example, if a chart is selected, the queries and results that it relies upon are included in the imported document.

9 **Click OK.**

After the import process is complete, a report detailing the imported sections is displayed. Dashboard Studio sections in the document being imported are ignored, and all other selected sections are imported. Imported sections with names of original sections are renamed (for example, *Query* becomes *Query2*).

10 **Optional: To examine the imported document in Interactive Reporting Studio, click View.**

Clicking View enables the imported document to be checked and connected to Dashboard Studio.

Importing Many to One

Import all non-Dashboard Studio sections from a folder that contains Interactive Reporting documents and combine them with a Dashboard Studio template to create one new ESM file.

➤ To import sections by using Many to One:

1 **From Import Method. select Many To One.**

2 **Click 📂 (next to Template), to locate the template.**

Select a Dashboard Studio template that provides the sections to be imported into the Interactive Reporting document to create an ESM. The Template box is pre-filled with the location of the template selected on the Start window.

3 **Click 📂 (next to Document Folder to Import), to locate the folder that contains the Interactive Reporting documents.**

The original, selected Interactive Reporting documents are not altered.

4 **Click 📂 (next to Save Path), to locate where the imported document or template is to be saved.**

5 **Click Import.**

After the import process is complete, a report detailing the imported sections is displayed. Dashboard Studio sections in the documents being imported are ignored, and all other selected sections are imported. Imported sections with names of original sections are renamed (for example, *Query* becomes *Query2*).

6 **Optional: To examine the imported document in Interactive Reporting Studio and connect it to Dashboard Studio, click View.**

Importing Many to Many

Import a folder that contains Interactive Reporting documents and combine each document with the same Dashboard Studio template to create a folder of matching imported ESM files.

➤ To import sections by using Many to Many:

1 **From Import Method, select Many To Many.**

2 **Click 📂 (next to Template), to locate the template.**

Select a Dashboard Studio template that provides the sections to be imported into the Interactive Reporting documents to create ESMs. The Template box is pre-filled with the location of the template selected on the Start window.

3 **Click 📂 (next to Document Folder to Import), to locate the folder that contains the Interactive Reporting documents.**

The original, selected Interactive Reporting documents are not altered.

4 **Click 📂 (next to Save Path), to locate where the imported documents or templates are to be saved.**

5 **Click Import.**

After the import process is complete, a report detailing the imported sections is displayed. Dashboard Studio sections in the documents being imported are ignored, and all other selected sections are imported. Imported sections with names of original sections are renamed (for example, *Query* becomes *Query2*).

6 **Optional: To examine the imported documents in Interactive Reporting Studio and connect them to Dashboard Studio, click View.**

The result is identical numbers of Interactive Reporting documents in the specified save path and in the specified Document Folder to Import folder.

Using the Merge Tab

Using the Merge tab, you combine sections in two ways:

- Merging Two to One
- Merging Many to One

Merging Two to One

When you merge two Interactive Reporting documents, the merged document contains all specified sections from both documents.

➤ To merge sections by using Two to One:

1 **From Merge Method, select Two To One.**

2 **Optional: Select Select and Re-order Sections.**

Only sections from the secondary document can be selected. All sections from the primary document are included in the merged document.

3 **Optional: Select Remove duplicate images from the final document.**

See “Remove Duplicate Images from the Final Document” on page 117.

4 **Click 📂 (next to Primary Document), to locate the primary Interactive Reporting document to merge.**

Primary documents take precedence over secondary documents. Merged secondary sections with names of primary sections are renamed (for example, *Query* becomes *Query2*). The primary document remains unchanged.

5 **Click 📂 (next to Secondary Document), to locate the secondary Interactive Reporting document to merge.**

When two Dashboard Studio documents are merged, all hidden sections in the secondary document must be removed before the merge process begins. To remove them, select View, then Unhide Section, unhide each section, and delete the sections from Sections.

Unless specific requirements are stipulated; for example, the need to merge a Dashboard Studio section, it is recommended that Import be used to create Dashboard Studio documents.

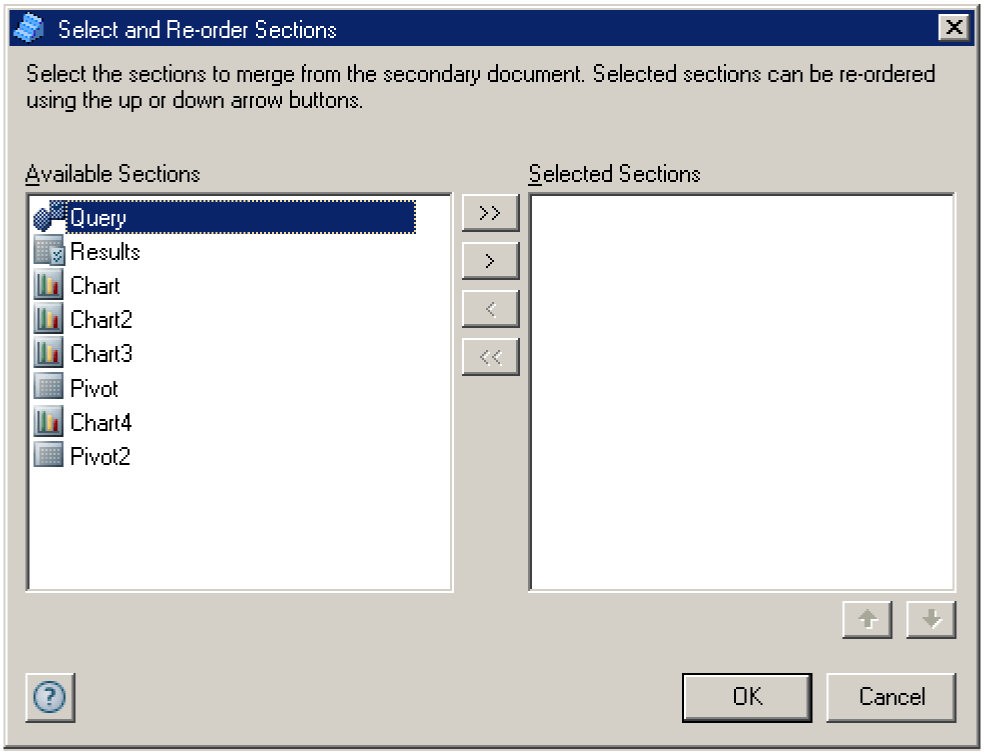
6 **Click 📂 (next to Save Path), to change the default location or the name of the merged document.**

Save Path points to the primary document. Dashboard Studio Merge Utility uses the primary document name (with *_merged* appended to the end) as the merged document default name.

If the generated file name duplicates an existing file name, specify another name.

7 **Click Merge.**

Select and Re-order Sections is displayed, if the option was selected on the Merge tab.



Select the sections to merge from the secondary document. Selected sections can be re-ordered using the up or down arrow buttons.

8 **If Select and Re-order Sections is displayed, perform an action to move sections between Available Sections and Selected Sections:**

- Select one or more sections, and click > or <
- Click >> or << to move all sections
- Double-click a section

All dependent sections are also selected and moved.

9 **Optional: Reorder Selected Sections by selecting a section and clicking ↑ or ↓.**

Queries are processed in the order in which they occur in the document. If the data returned by one query is required by another, the ability to reorder queries is useful.

10 **Click OK.**

After the merge process is complete, a report detailing the sections that were merged is displayed.

11 **Optional: To examine the merged Interactive Reporting document and connect it to Dashboard Studio, click View.**

Merging Many to One

Merge multiple Interactive Reporting documents from one folder into one merged document. The merged document contains all sections from the original documents.

➤ To merge sections by using Many to One:

1. **From Merge Method, select Many To One.**
2. **Click 📂 (next to Source Document Folder), to locate the folder that contains the documents to be merged.**

   If Source Document Folder contains ESMs, it is recommended that Import be used to avoid duplication of hidden sections. Otherwise, all hidden sections from each ESM (except for the first document in the folder) must be removed.

3. **Click 📂 (next to Save Path), to locate where the merged document is to be saved.**

   If the specified file name duplicates an existing file name, specify another name.

4. **Click Merge.**

   After the merge process is complete, a report detailing the merged sections is displayed.

5. **Optional: To examine the merged Interactive Reporting document and connect it to Dashboard Studio, click View.**

Note:

When merging components into Dashboard Studio templates or Interactive Reporting documents, both BMP and GIF files are available.

6

Using the Dashboard Development Services Update Utility

In This Chapter

About Dashboard Development Services Update Utility

The Dashboard Development Services Update Utility updates JavaScript in Interactive Reporting documents, provided that the JavaScript is designed to create a layered architecture using dashboard sections.

- **Layer 1**—Data and views (queries, charts, pivots, and reports)
- **Layer 2**—Sections with which you interact; charts, pivots, and user interface controls (lists, buttons, drop-down lists, and so on that contain very simple, one-line calls to Library routines)
- **Layer 3**—Sections that contain reusable JavaScript functions, classes, and components

The Dashboard Development Services Update Utility updates only the contents of layer 3. It uses a `newsections.bqy` document (an Interactive Reporting document that contains the latest layer 3 sections) to *push* new layer 3 sections into *old* dashboards, thus converting *old* dashboards into *new* dashboards.

Within documents that are being updated, sections that are common to the current document and the `newsections.bqy` document are replaced by sections from the `newsections.bqy` document. As of Release 8.3.1, release information within sections is used to ensure that *earlier* sections do not replace *later* sections. Therefore, as of Release 8.3.1, a section is replaced only if its corresponding `newsections.bqy` section references a more recent release.

In summary, the update rests on these assumptions:

- JavaScript is developed in layers
- A layer is one or more Interactive Reporting document sections that together implement some discrete function or set of related functions

- Document scripts are treated as if they are sections with `onStartUp`, `onShutDown`, `onPreprocess`, `onPostProcess` events

Unicode Functionality

The Dashboard Development Services Update Utility automatically converts old code page based Interactive Reporting documents to Unicode before use, enabling documents with different languages to be updated.

Tip:

If running a non-Unicode Interactive Reporting document through the Dashboard Development Services Update Utility results in an error, open and save the non-Unicode document in Interactive Reporting Studio before an update.

Consolidate Images in Resource Manager

The utility automatically removes duplicate and unused images from the Resource Manager. See “Resource Manager” in the *Hyperion Interactive Reporting Studio User's Guide.*

Update Workflow

The workflow to update or transform sections is described in this topic.

1. Create or update `newsections.bqy`. A `newsections.bqy` is provided with the installation (see “New Sections File” on page 126).
2. Configure `JavaScriptUpdateConfig_dds.js`. A `JavaScriptUpdateConfig_dds.js` is provided with the installation (see “Configure Configuration Files” on page 127).
3. Update documents (see “Updating Documents” on page 127).

Note:

Steps 1 and 2 are optional. If templates are not and will not be customized, proceed directly to step 3 to update documents.

New Sections File

The Dashboard Development Services Update Utility uses `newsections.bqy` as its input. It is an Interactive Reporting document that contains the latest version of the infrastructure (the shared JavaScript function and object constructors). The utility opens each Interactive Reporting document in a nominated list and performs a compare operation which checks if section names from the list exist in `newsections.bqy` and the Interactive Reporting document to be updated.

If a section exists in both, the section in the Interactive Reporting document is removed and replaced with the section from `newsections.bqy`.

Configure Configuration Files

The configuration file contains a set of JavaScript functions called by the update script at crucial points in the process. The configuration file enables the update process to be refined or customized to suit your requirements. Together with the new sections file, this configuration file updates Dashboard Studio templates or documents.

The Dashboard Development Services Update Utility enables you to move values from the contents of text labels, drop-down lists, and lists from the old dashboard sections that are to be discarded, and copy them into the equivalent shapes of the new sections when they are inserted into the Interactive Reporting document. The configuration file is used to identify the text labels, drop-down lists, and lists in the sections whose values are to be transferred when updating a file. Values to be maintained must be specified in this file.

Refer to the JavaDoc within the configuration file on how to customize the configuration file and the update process.

Two configuration files are provided with the installation; `JavaScriptUpdateConfig_dds.js` and `JavaScriptUpdateConfig_blank.js`. You can customize either file, however, `JavaScriptUpdateConfig_dds.js` works specifically with Dashboard Studio.

Modify the Configuration File

The configuration file can be modified to include custom components, and to add or remove sections.

The Dashboard Development Services Update Utility compares the available sections in the documents to update and the specified new sections file. If a section exists in `newsections.bqy` and the Interactive Reporting document to be updated, the corresponding section from the new sections file replaces the section in the Interactive Reporting document. The configuration file enables you to specify sections that are to be added to the documents to update even if these sections did not previously exist. Similarly, you can specify sections to be removed from the documents to update.

Updating Documents

The Dashboard Development Services Update Utility enables you to update documents in one of three ways:

- Using the Update One Method—Update one Interactive Reporting document at a time with a GUI
- Using the Update Many Method—Update a folder of Interactive Reporting documents with a GUI

- Selecting Documents to Update with Command Line Updates

Using the Update One Method

Update one Interactive Reporting document at a time using the utility GUI.

➤ To update one Interactive Reporting document:

1. **Open Dashboard Development Services Update Utility.**
2. **From Update Method, select Update One.**
3. **Select the preferred backup option.**

   - Selecting Place Updates in the Update Folder, creates an Update folder in the source path and the updated document is saved to the folder (If a document with an identical name exists in the Update folder, a timestamp is added to the document currently being updated. The original document in the Update folder is not overwritten)
   - Selecting Place Originals in Backup Folder, creates a Backup folder in the source path (The original document is saved to the Backup folder with a timestamp added to the file name. The updated document is saved to the source path and takes the name of the original document)

   When the option to create a backup of the document is selected, the original document must not be set to read-only.

4. **Click 📂 (next to JavaScript Configuration File), to locate the configuration file.**

   The JavaScript configuration file determines the sections to replace, add, delete, or preserve in the specified document to update. A configuration file is provided by default.

   Note: To add the Layout Manager to updated documents, click **Browse** and select `JavaScriptUpdateResizeOn_dds.js` from the `\DDS\scripts\DDSUpdate` directory as the configuration file.

5. **Click 📂 (next to New Sections File), to locate** `newsections.bqy`**.**

   A new sections file is provided by default, which contains the latest version of the infrastructure sections.

6. **Click 📂 (next to Document to Update), to locate the document to update.**

   The save path of the updated document is generated in Save Path of Updated Document(s).

7. **Click Update.**

   When the update process is complete, a report confirms a successful or unsuccessful update.

8. **Click View to launch the updated document in Interactive Reporting Studio.**

Using the Update Many Method

Update multiple Interactive Reporting documents using the utility GUI.

➤ To update a folder of Interactive Reporting documents:

1. **Open Dashboard Development Services Update Utility.**
2. **From Update Method, select Update Many.**
3. **Select the preferred backup option.**

   - Selecting Place Updates in the Update Folder, creates an Update folder in the source path and the updated documents are saved to this folder (If a document with an identical name exists in the Update folder, a timestamp is added to the document currently being updated. The original document in the Update folder is not overwritten)
   - Selecting Place Originals in Backup Folder, creates a Backup folder in the source path (The original documents are saved to the Backup folder with a timestamp added to the file name. The updated documents are saved to the source path and take the name of the original documents)

   When the option to create a backup of the documents is selected, the original documents must not be set to read-only.

4. **Click 📂 (next to JavaScript Configuration File), to locate the configuration file.**

   The JavaScript configuration file determines the sections to replace, add, delete, or preserve in the specified document to update. A configuration file is provided by default.

   Note: To add the Layout Manager to your updated documents, click **Browse** and select `JavaScriptUpdateResizeOn_dds.js` from the `\DDS\scripts\DDSUpdate` directory as the configuration file.

5. **Click 📂 (next to New Sections File), to locate** `newsections.bqy`**.**

   A new sections file is provided by default, which contains the latest version of the infrastructure sections.

6. **Click 📂 (next to Document Folder to Update), to locate the folder to update.**

   The save path of the updated documents is generated in Save Path of Updated Document(s).

7. **Click Update.**

   When the update process is complete, a report confirms a successful or unsuccessful update.

8. **Click View to open the updated folder.**

Tip:

If Interactive Reporting Studio does not launch properly after an Update Many operation in the Dashboard Development Services Update Utility, open the Windows Task Manager and end any `brioqry.exe` process before trying to launch the document.

Command Line Updates

The Dashboard Development Services Update Utility gives you the option of updating documents with a command line. The major advantages of this method include:

- Multiple documents from different locations can be updated simultaneously
- A permanent list of documents to update can be built

Dashboard Development Services Update Utility performs generic transformations based on a specified script. Generic transformations are only available using the command line. Any customized script can be run from the command line.

Selecting Documents to Update

Use scripts that are provided or customized scripts to perform the transformation.

➤ To select files or folders to update using the command line:

1 **Navigate to the bin folder.**

If the installation was not customized, the bin folder is under `C:\Hyperion\products\biplus\bin`. However, if the installation was customized, fix the path to be relative.

2 **Select `runTransformScript.bat`, right-click, and select Edit.**

This enables you to specify the script, parameters, and options to run.

The BAT file includes these lines:

```
set PARAM=%PARAM% -js ""
set PARAM=%PARAM% -param:name=""
```

The first line is the script to run and the second line is a parameter placeholder.

3 **Specify the script.**

The script to run must be entered after the `-js` option in the BAT file. For example, `-js "JavaScriptUpdate.js"`.

4 **Specify the script parameters using one of these two methods:**

A script can have many or no parameters.

a. Using the `-param` option.

All \ and " must be escaped.

The `-param` option has this syntax for single values:

```
-param:name="value"
```

The `-param` option has this syntax for an array with two items:

```
-param:name="[\"value\", \"C:\\Hyperion\\products\\biplus\"]"
```

The first item is value and the second item is `C:\Hyperion\products\biplus\`.

For example, JavaScript Update requires these parameters:

- The configuration file (parameter name: updateConfig)
- The new sections file (parameter name: newSectionFile)
- The document or file to update (parameter name: targetBQYs)

- The update folder (parameter name: updateFolder) OR the backup folder (parameter name: backupFolder)

An example:

```
set PARAM=%PARAM% -js "C:\\Hyperion\\products\\biplus\\DDS\\scripts\
\DDSUpdate\\JavaScriptUpdate.js"
set PARAM=%PARAM% -param:updateConfig="C:\\Hyperion\\products\\biplus\
\DDS\\scripts\\DDSUpdate\\JavaScriptUpdateConfig_dds.js"
set PARAM=%PARAM% -param:newSectionFile="C:\\Hyperion\\products
\bipPlus\\DDS\\scripts\\DDSUpdate\\newsections.bqy"
set PARAM=%PARAM% -param:targetBQYs="C:\\Folder or Document to update"
set PARAM=%PARAM% -param:updateFolder="C:\\Update folder"
```

Optional: Replace the last line, if creating a backup folder:

```
    set PARAM=%PARAM% -param:backupFolder="C:\\Backup folder"
```

b. Using the `-batch` option.

This option enables you to execute a script, multiple times with different parameters. Each line in the batch file represents one execution of a script. The parameters are comma-separated; for example, `name1="value1", name2="value2"`.

This example is an equivalent batch file for the JavaScript Update example in step 4a. It is an example of the contents of a BAT file:

```
   updateConfig="C:\\Hyperion\products\biplus\\DDS\\scripts\\DDSUpdate
\\JavaScriptUpdateConfig_dds.js", newSectionFile="C:\\Hyperion
\products\biplus\\DDS\\scripts\\DDSUpdate\\newsections.bqy",
targetBQYs="C:\\Folder to update", updateFolder="C:\\Update folder"
```

Use this code to point to the BAT file:

```
set PARAM=%PARAM% -js "C:\\Hyperion\products\biplus\\DDS\\scripts\
\DDSUpdate\\JavaScriptUpdate.js"
set PARAM=%PARAM% -batch "C:\\Hyperion\products\biplus\\DDS\\scripts\
\DDSUpdate\\batchFile.txt
```

`BatchFile.txt` is used as an example name for the parameter file.

5 **Optional: To include an instruction in the BAT file, to fail on error (foe), add the parameter:**

```
set PARAM=%PARAM% -foe
```

If this parameter is included and the file encounters an error when running the transformation, the process will stop and an error message will be displayed in the log. If the line is not included, any errors that are encountered are skipped. The log alerts you to errors. This works only for JavaScript Update.

6 **Optional: To include an instruction in the BAT file, to create a report, enter this line:**

```
set PARAM=%PARAM% -rep "C:\\Hyperion\\products\\biplus\\DDS\\report"
```

A file path and folder must be specified; for example, `C:\Hyperion\products\biplus\DDS\report`.

7 **Save and close the BAT file.**

8 **In Interactive Reporting Studio, open a Command window.**

9 **Navigate to the bin folder.**

For example, navigate to `C:\Hyperion\products\biplus\bin`.

10 **Run** `runTransformScript.bat`**.**

The BAT file must be run from the bin directory.

The Command window displays output. If successful, a message is displayed.

7 Updating Documents with Advanced Scripting

In This Chapter

Customizing Scripts

This topic discusses customizing scripts to update documents in EPM Workspace or on the desktop in Interactive Reporting Studio.

EPM Workspace Custom Scripting Environment

The custom scripting environment of the Oracle's Hyperion® Impact Management Services provides a mechanism for manipulating the content and structure of an Interactive Reporting document through a Document Object Model (DOM) and JavaScript. Although this environment is similar to the scripting environment in Interactive Reporting Studio, there are differences. For example, the custom scripting environment of the Impact Management Services:

- Does not work in the context of an active Interactive Reporting document
- Provides access to all properties in the document
- Does not perform logical system-level integrity checks
- Is not contained inside the Interactive Reporting document
- Executes a script over multiple documents

The custom scripting environment performs arbitrary, common transformations on one or more documents. This mechanism is used to implement the Update Data Models and Update JavaScript features of the Impact Management Services.

Scripts can be imported into EPM Workspace and then run using the Custom Update feature of the Impact Management Services to make changes to other imported documents. These scripts can also be run on a desktop by the Dashboard Development Services Update Utility. From the desktop, changes can only be made to files on disks visible from that desktop. The desktop is typically a development and test environment.

Scripts in EPM Workspace run under the control of the Impact Management Services and consequently can use the Undo feature. If a change made through scripts is unwanted, the task that used the script can be undone and the documents are returned to the pre-script state.

See "Using Impact Management Services" in the *Hyperion Workspace Administrator's Guide.*

Calling Scripts

The Impact Management Services scripts can be executed within EPM Workspace or on a client desktop in Dashboard Development Services Update Utility.

Calling Scripts in EPM Workspace

Within EPM Workspace, the Custom Update feature of the Impact Management Services is used. The feature presents three steps to execute scripts:

1. Browse for and select a script.
2. Enter parameters required by the script. The Impact Management Services builds a parameter form that is specific to that script. Or you can specify sets of parameter values by using a batch input file.
3. **Optional:** Schedule when to execute the script.

Calling Scripts in Dashboard Development Services Update Utility

Use the Dashboard Development Services Update Utility to execute scripts using batch files; for example, `runTransformScript.bat`. Each script requires a specific batch file containing the parameters required by the script. See "Selecting Documents to Update" on page 130.

The JavaScript Update transformation replaces earlier sections that contain JavaScript with later versions of sections. Property settings from the earlier section are transferred to the later section, so the later code can work with earlier properties. The Dashboard Development Services Update Utility supports both batch and interactive mode. See "Updating Documents" on page 127.

Monitoring Script Execution

The Show Task Status list in EPM Workspace enables progress monitoring of script execution. While awaiting execution, and during the running of a script, the status is displayed as Waiting (gray). Upon completion, the status changes to Success (green) or Fail (red).

When a task is complete, double-clicking the entry in the Show Task Status list displays generated log messages. Use logs to debug errant scripts.

In Interactive Reporting Studio, logging is written to the file called *`dds.log`*. In a standard installation, the logs folder is located under `C:\Hyperion\products\biplus\logs`.

Custom Scripts

These scripts are available to update documents in EPM Workspace or Interactive Reporting Studio.

JavaScriptUpdate.js

The JavaScriptUpdate script enables users to take advantage of the latest dashboard features without having to re-create documents from scratch. See "Using Impact Management Services" in the *Hyperion Workspace Administrator's Guide.*

UpdateDataModels.js

The UpdateDataModels script enables data models in documents to be updated to reflect changes in underlying databases. See "Using Impact Management Services" in the *Hyperion Workspace Administrator's Guide.*

SortDataModelTopics.js

The SortDataModelTopics script enables documents to be updated so the topics in data models are displayed in EPM Workspace in a user-defined order or alphabetically.

When an Interactive Reporting document is opened in EPM Workspace and a query is visible, a list of topics is displayed in the catalog pane under Tables. The topics are displayed in the order in which they were added to the Interactive Reporting document which makes locating topics difficult if there are many in the list.

The SortDataModelTopics script enables the user to specify the order in which the topics are displayed in these lists, using three parameters.

There are two ways to specify the sort order:

1. Use the first parameter to select a file containing a list of topic names, in the order preferred by the user.

2. Use the second parameter (true or false) to specify whether topics that are not included in the sorted file should be ordered alphabetically.

Topics that are not mentioned in the sorted file are placed after topics that are mentioned, and are ordered according to the second parameter. Therefore, if you provide an empty file and the second parameter is true, all topics will be ordered alphabetically, making it easy to locate a topic in the list.

Note:

The empty file should contain a blank line.

The third parameter enables selection from a set of files to be updated, through a multi-file picker.

A version is added for each successfully updated file. Therefore, double-clicking a file in EPM Workspace displays the updated content.

RevertImageResources.js

The RevertImageResources script is desktop-specific, and is not used in EPM Workspace.

A Resource Manager centralizes image content storage. For Interactive Reporting documents created in earlier releases, duplicate images can be optionally merged for efficiency.

A side effect of this merging is that if a pre-9.3 Release document is opened, optimized, and saved in a 9.3 or later release, the document loses its images when opened in an earlier release of Interactive Reporting Studio, since the relocation and rationalization of the images is not understood by earlier releases.

To retain compatibility with earlier releases in practical terms, the RevertImageResources script provides a facility to undo the relocation and image merging and return the Interactive Reporting document to the pre-9.3 format.

Caution!

The script is only useful with documents saved using Interactive Reporting Studio Release 9.3 or later, after the new image consolidation and translation features have been used. See "Consolidate Images in Resource Manager" on page 126.

Running the RevertImageResources Script

The RevertImageResources script can be run using `runRevertImageResourcesScript.bat`. If the installation was not customized, `runRevertImageResourcesScript.bat` is located in `C:\Hyperion\products\biplus\bin`.

This script accepts a path as a command line parameter:

- This can be the absolute path to an Interactive Reporting document to revert it to its pre-9.3 release format
- This can be the absolute path to a folder of Interactive Reporting documents to revert them all to their pre-9.3 release format

Note:

Any backslashes (\) in the path should be duplicated. For example, `C:\\docs\\mydocument.bqy`.

See "Using the RevertImageResources Script on a Folder" on page 18 and "Using the RevertImageResources Script on a File" on page 18.

Script Parameters

The parameters required by a script are specified using comments in the header. These are similar in format to the JavaDoc comments used to document Java.

The minimum that can be specified to define a parameter is the name; for example, @param sourceLanguage.

This assumes that the input is a simple string and displays an (initially empty) text box on the UI.

Optional: An @inputType line enables more specific data input methods:

- text—Text
- password—Text displayed as asterisks (*)
- file_picker_single_value—Select one file from the repository
- file_picker_multi_values—Select multiple files from the repository, all of which constitute one value
- file_picker_multi_values_parallel_execution—Select multiple files from the repository, all of which can be processed in parallel by separate instances of the script
- dropdown—Select from a predefined set of fixed values

Input types can be given a default value using @defaultValue. The @defaultValue of file_picker type is the fully qualified path and name; for example,

`/Administration/Impact Manager/Script Repository/SortDataModelTopics.js.`

Note:

If this is not unique or the file does not exist, then a warning dialog is displayed and the parameter default value is not set. It has the same effect as not specifying the default value.

Drop-down lists require a separate @comboValues line that specifies possible choices, separated by commas.

Note:

For custom scripts, parameter values are only validated when the script is executed, not at submission time. For example, if an unacceptable value is specified for a script, the user is not informed at the time of task submission. If a script cannot recover from invalid data, it logs a message and throws an exception, causing the status to display as Fail (red) in Show Task Status, alerting the user to the problem.

Logging

Scripts use log messages to communicate with users. In EPM Workspace, logs are accessed through the Show Task Status list. On the desktop, in Interactive Reporting Studio, users view `dds.log`, which is the default name for script log files, and can be changed in the BAT file. In a standard installation, logs are located in the default folder `C:\Hyperion\products\biplus\logs`. Each execution of the script clears the log file. On the desktop, the log represents the entire set of tasks one for the folder and one for each file.

Note:

In the Dashboard Studio Inspector Utility Custom Update Script feature logging messages are also displayed in Task Options, in the Results window. See Chapter 8, "Dashboard Studio Inspector Utility."

The higher the level set, the more messages are displayed in the logs. The levels are explained in Table 9.

Table 9 Logging Levels

Level	Description
Debug	Determines what is happening during script development or to track down problems
Warn	Warns of recoverable problems that require correcting
Error	Indicates the inability to correctly perform the requested processing
Fatal	Indicates the script cannot continue
Always	Messages that are always displayed

There are env methods available to log messages at each of these levels. For example, env.logInfo(), env.logDebug(), and so on. See "ScriptEnvironment Object" on page 146.

There is also a default logging level associated with the script execution. The env.log() method logs messages at that level. The default level is initially debug, but can be changed by using env.setLogLevel().

The env.logClassName() method provides information on the type of an object, because values returned by methods are a combination of JavaScript and Java objects.

Writing Document Information into the Log

A simple script that works with an Interactive Reporting document or a folder of Interactive Reporting documents can be used to determine each file name and the number of sections it contains.

Example: Using a Windows command file for desktop execution

`Rhino.bat` is a general purpose command file that is used to launch most Impact Management Services scripts on the desktop. Parameters must be entered in the form: `<documentOrPath> param_1=value_1 param_2=value_2`.

Example: Using a script file for one file or a folder of files

Where the `document` parameter of the script is a folder, then the script instructs the environment to expand the folder into a list of files and calls the script once for each file in the list. Script execution is called a *task* and is monitored in EPM Workspace using the Show Task Status and Task Management lists. If the `document` parameter of the script is one file, then the script is called once.

Document Object Model Tree Structure

The Document Object Model (DOM) is a tree structure of nodes and properties; each node is made up of more nodes and properties. The DOM and JavaScript engine provide the ability to interrogate and update the state. In Impact Management Services, it is not necessary to expand the whole Interactive Reporting document, only those nodes with properties of interest. For example, when doing a data model update, only query and data model sections need to be expanded. However, this procedure requires no expansion, as the information is available at the top level of the DOM.

Expanding part of an Interactive Reporting document speeds up document loading and consumes less memory. The document loading routines expand only what is required as it is requested. Any scripts that make use of this optimization continue to work; the Document Conversion Strategy parameter is ignored.

Note:

You can include the `bqReadWriteDom` and `bqReadOnlyDom` scripts; however, their values are ignored.

Each document manipulated by a script is stored in the form of a DOM, represented by a tree of nodes, each of which contains a set of associated properties.

The DOM for a document is acquired by retrieving the file and loading the content; for example,

```
var uuid = env.getParameterValue("document");
var file = repository.retrieveFile(uuid);
var dom = env.getBqyDocument(file, bqReadWriteDom,
bqDashboardReportStrategy)
```

The first line retrieves the parameter that contains the document UUID. The second line is used to copy the file from the repository to a temporary file which is deleted when the script ends. The third line loads the content of the file, providing a BqyDocument object that represents the DOM.

Note:

The second parameter, bqReadWriteDom, specifies that the document is to be rewritten. If it is not to be rewritten, specify bqReadOnlyDom to reduce the amount of memory required for the DOM. The third parameter is the document conversion strategy, bqDashboardReportStrategy. It determines how much of the underlying document structure is accessible to the script.

Using different strategies, the amount of memory required by a script can be reduced, as can the time spent loading the document.

Document Conversion and Loading Strategies

When loading documents, you can save memory by loading only those portions of the DOM that are required by a given script, For example, JavaScript Update uses only dashboard sections. To log a list of section names in a document, you do not need to load the entire tree of nodes that lie beneath the sections. An example of the required syntax:

```
env.getBqyDocument(documentFile, bqReadWriteDom,
bqJavascriptUpdateStrategy);
env.getBqyDocument(documentFile, bqReadOnlyDom, bqDashboardReportStrategy)
env.getBqyDocument(documentFile, bqReadOnlyDom, bqDashboardReportStrategy)
```

These strategies are provided for loading sections:

- bqDashboardReportStrategy—Only dashboards and reports
- bqDatamodelUpgradeStrategy—All data models and queries
- bqJavascriptUpdateStrategy—Only dashboards
- bqTopLevelSectionsStrategy—All sections and section level properties (a minimal DOM is created)
- null—The whole document

Note:

A just-in-time approach to DOM buildingmakes document loading strategies redundant. Any strategy parameter provided is simply ignored. In addition, the bqReadWriteDom script is ignored.

Traversing the Document Object Model

To manipulate the content of a node in the DOM, you must locate the node.

The top-level nodes that represent the sections of a document can be accessed directly by using the Sections collection. The shapes within a dashboard are accessible through its Shapes collection. However, there is no collection for the children of a node.

Methods are provided to access the children of a node:

- getChildren()—Returns a complete list of children of a node
- getChildrenOfType()—Returns a list of children of a node that have a specific type
- addChild()—Adds a new child to the end of a list of children of a node
- removeChild()—Removes the specified node from a list of children of a node
- setChildren()—Replaces a list of children of a node with another list
- dump()—Dumps the DOM tree, starting at the given node, for debugging

To iterate over all subnodes of a given node, use getChildren() to retrieve a list that contains them. Use getChildrenOfType() to limit this to the children of a particular type. For example, the Root.MyDocument node contains a Rpt.DocComp node for each section in the document, which can be located using this line:

```
var sections = root.getChildrenOfType("Rpt.DocComp");
```

A node is added as a child of another by using addChild(). Use this to copy a node from one part of the DOM (or the DOM of another document) to another location.

To remove a child node, use removeChild().

Note:

The list of children returned by getChildren() and getChildrenOfType() is read-only. If you update the list by assigning a new value to an entry, this does not affect the node. However, the current list of nodes can be replaced using setChildren().

The content of a sub-tree of the document can be written to the log using dump(). By default, this dumps the tree to standard output, but by supplying parameters, it can be written to any print stream.

XPath-Style Searching

While obtaining lists of child nodes enables access to the entire DOM, you can search for nodes that satisfy a set of criteria.

For example, this code can be used to log the names of all shapes within a document:

```
for (var i = 0; i < dom.Sections.length; i++) {
    var section = dom.Sections[i];

    if (section.Type == bqDashboard) {
        env.log("Dashboard " + section.AnnotName + " has shapes");

        var shapes = section.Shapes;
        for (var j = 0; j < shapes.length; j++)
                env.log(shapes[j].Name);
    }
}
```

The DOM provides user-friendly collection names for both the Sections inside a document and the Shapes inside a dashboard. However, a complex search example that looks for all DataThreshold.DataThreshold nodes inside all ThreshFmt.ThreshFmt nodes, inside all ColColl.Item nodes, inside all table sections, results in multiple nested loops.

The Impact Management Services scripting provides an alternative approach, through XPath-style searches. For example, this is the code to use for the complex search example:

```
var items = dom.findNodesByPattern("/BQY/Root.MyDocument/Rpt.DocComp"
        + "/ColColl.Item/ThreshFmt.ThreshFmt"
        + "/DataThreshold.Threshold");
```

This single statement provides an array that contains the required nodes. Property matching requirements can be included to narrow down which nodes are to be returned.

For example, to limit the result to those nodes in the column named *Drawn* inside the table named *Rankings*.

```
var items = dom.findNodesByPattern("/BQY/Root.MyDocument"
        + "/Rpt.DocComp[AnnotName='Rankings']"
        + "/ColColl.Item[Label='Drawn']/ThreshFmt.ThreshFmt"
        + "/DataThreshold.Threshold");
```

Searches need not begin at the root of the DOM. If a variable that contains a section node is searched, use a relative path to find other nodes beneath that section; for example:

```
var table = dom.findNodesByPattern("/BQY/Root.MyDocument"
        + "/Rpt.DocComp[AnnotName='Rankings']");

var items = table.findNodesByPattern("ColColl.Item[Label='Drawn']"
        + "/ThreshFmt.ThreshFmt/DataThreshold.Threshold");
```

Use getNodesByPattern() if there is a possibility that a node may not exist (that is, if documents to be processed using a script may not contain this node) or where there can be many of these nodes. In these cases, the length of the returned array is used to determine the situation.

However, if one node matching the pattern is guaranteed, use getNodeByPattern() which returns one object, rather than an array.

The search mechanism provides two wildcard facilities. An asterisk (*) in place of a node name, represents any type of node. A pair of slashes (//) represents any number of intervening nodes.

For example, to find all images in a document, in dashboards or reports (in the body, header, footer, section header, or section footer), use this example code:

```
var pattern = "//Box.Item[RuntimeClassName='PictField']";
var pictures = dom.findNodesByPattern(pattern);
```

Differences Between the Impact Management Services and Interactive Reporting Studio Document Object Models

Impact Management Services provides the DOM to its scripts and Interactive Reporting Studio, including Oracle's Hyperion® Interactive Reporting Web Client, provides the BQY object model (BOM) to scripts embedded within Interactive Reporting documents.

The DOMs available in the Impact Management Services scripting differ from those provided to event handlers in Interactive Reporting Studio scripting:

- All collection indices start at zero, rather than one
- The node names and properties match those stored in the underlying document, as displayed in the Dashboard Studio Inspector Utility
- The BOM provides user-friendly names to resemble the view through the Interactive Reporting Studio; whereas, the DOM provides fewer user-friendly names
- The BOM does not provide access to the majority of properties, however the DOM provides access to all properties
- Using the DOM, the BOM event handlers for sections and shapes cannot be called to effect changes to the document
- The BOM provides safety checks and restrictions, however the DOM provides only basic type checking
- Using the DOM, you can change and transform anything; for example, you can create files that are not recognized by other software

Investigating the Impact Management Services DOM Structure

The Dashboard Studio Inspector Utility, included with the Interactive Reporting Studio installation, provides an explorer-style view of the DOM. See Chapter 8, “Dashboard Studio Inspector Utility.”

The left pane displays the nodes contained within the document as a tree. The right pane displays the names of all properties of the selected node, and their current values and associated data types.

Use the Inspector Utility when writing scripts, to determine where in the DOM the data resides that must be manipulated to achieve the intended change, and to generate the path that provides programmatic access to data of interest.

Accessing Properties

Access properties in Impact Management Services, as you do in Interactive Reporting Studio. The only difference is the DOM exported by Interactive Reporting Studio provides more user-friendly names for frequently used properties.

To learn the names of properties associated with a node in the DOM use the Dashboard Studio Inspector Utility. See Chapter 8, "Dashboard Studio Inspector Utility."

For example, this code accesses the IsHidden property of a dashboard, making the section visible if it is hidden.

```
var dashboard = dom.Sections["Dashboard"];

if (dashboard.IsHidden) {
        env.log("Making section " + dashboard.Name + " visible");

        dashboard.IsHidden = 0;
}
```

Collections

An important difference between the Interactive Reporting Studio scripting DOM and the Impact Management Services DOM is that all collections are zero-based, not one-based. For example, a loop that would have been coded as:

```
for (var i = 1; i <= collection.Count; i++)
    // some processing on collection[i]
```

is now written as:

```
for (var i = 0; i < collection.length; i++)
    // some processing on collection[i]
```

Property Types

Every property of a DOM node has one of these data types:

- Byte
- DWord
- Long
- String
- Structure
- Word

Accessing the File System

To access the underlying file system from within a script; for example, where a large amount of configuration information is needed that does not change from execution to execution, use these methods.

- env.getFileSystem()—Retrieve an object that provides access to the underlying file system
- env.createTempFile()—Create a temporary file that is cleaned up when the script completes

- fs.getFile()—Retrieve a Java File object that refers to the file with a given path within EPM Workspace
- fs.writeBytesToStream()—Write the contents of a byte array to a file

General Java Code in Scripts

It can be necessary to construct Java objects as part of processing a script. For example, RevertImageResources creates a FileOutputStream using the call:

```
var fos = new Packages.java.io.FileOutputStream(imageFile);
```

The call is of the form:

```
var object = new Packages.java.some.package.ClassName(necessary,
parameters);
```

Using Batch Input Files

All parameters for a transformation script can be entered interactively by using the user interface, or you can request the processing of many sets of parameters by providing them as a batch input file.

Each line of a batch input file contains a complete set of parameters, as a comma-separated list of name="value" specifications.

For example, to use the SortDataModelTopics script to transform the three documents "/some.bqy", "/some/other.bqy" and "/yet/another/example.bqy", using the topic orderings in "/order.txt", and sorting unspecified topic names alphabetically, use this input file:

```
orderings="/order.txt",sortUnknownTopics="true",document="/some.bqy"
orderings="/order.txt",sortUnknownTopics="true",document="/some/other.bqy"
orderings="/order.txt",sortUnknownTopics="true",document="/yet/another/
example.bqy"
```

Note:

Each parameter value is quoted and all of them must be included on each line, even where the value does not change.

In EPM Workspace, the values of any parameters that represent files need to be UUIDs. The sample scripts are explicitly coded to enable batch files to specify file paths, by using code similar to this to convert them into UUIDs where necessary:

```
var document = env.getParameterValue("document");

if (document.substring(0, 1) == "/")
    document = repository.getFileUuid(document);
```

To enable annotation of batch input files, blank lines and any lines beginning with # are ignored.

Note:

The code also works on the desktop, because there the UUID of a file is identical to the file system path.

Scripting References

This topic includes scripting references and examples of objects, methods, and properties that are available to use on the desktop and in EPM Workspace.

ScriptEnvironment Object

Each script has a global variable called *env,* which provides access to the ScriptEnvironment object within which it runs and hosts the features that lead to granting access to documents and the document repository. A repository can be the Reporting and Analysis repository, if the script is running in EPM Workspace, or the file system, if the script is running on the desktop.

expandRequestAction()

Actions are added to the list for the task. Generally expandRequestAction() is used to generate multiple additional tasks to handle a collection of input files. For example, a user requests that an action occurs for a folder. The folder is expanded into an array of files and the script runs for each file in the array.

Example using expandRequestAction():

```
env.expandRequestAction(strParam, arrUuidValues)
```

Parameter	Description
strParam	Represents the file descriptor that is to be expanded
arrUuidValues	An array of the unique identifiers of the set of files on which to act. In EPM Workspace this is a set of real UUID values, on the desktop it is the list of paths of the files as expanded

getBqyDocument()

Used to retrieve a document from the repository, create the DOM, and provide access to the nodes and properties of the Interactive Reporting document.

Example using getBqyDocument():

```
var domBqy = env.getBqyDocument(filBqy, bqOpenMode, bqStrategy)
```

Parameter	Description
filBqy	Represents the Interactive Reporting document, generally retrieved using the retrieveFile method of the repository artifact

Parameter	Description
bqOpenMode	Defines the way the file is opened. For example, using bqReadOnlyDom the file is read-only, or using bqReadWriteDom the file has read/write properties. For scripts running in Release 9.3.1 or later this parameter is ignored because the Rhino engine loads nodes only if scripts attempt to reference them or the properties below them (see following table).
bqStrategy	Determines, as an efficiency measure, how much of the DOM is built. For scripts running in Release 9.3.1 or later this parameter is ignored because the Rhino engine loads nodes only if scripts attempt to reference them or the properties below them (see following table).

Property	Description
bqDashboardReportStrategy	Loads only dashboards and reports
bqDatamodelUpgradeStrategy	Loads only data models and queries
bqJavaScriptUpdateStrategy	Loads only dashboards
bqTopLevelSectionsStrategy	Loads all sections and the section level properties
null	Loads the whole document

getFileLines()

Used to retrieve file content from the repository as an array of strings, given the UUID.

Example using getFileLines():

```
var arrLines = env.getFileLines(filToRead)
```

Parameter	Description
filToRead	The text file from the repository to expand into the constituent lines. A file object consists of information about the file, but is not the content of the file.

getMimeTypeUuid()

Used to retrieve the UUID of the MIME type with the specified name.

Example using getMimeTypeUuid():

```
var uuiMimeType = env.getMimeTypeUuid(strMimeType)
```

Parameter	Description
strMimeType	The string representation of the MIME type to be returned; for example, application/x-brioquery

getParameterValue()

Values are obtained for a single-valued parameter, based on the name. If the named parameter value does not exist, return null.

The parameter value can be entered on the command line or through the Custom Update parameter gathering screen in EPM Workspace

Example using getParameterValue():

```
var strVal = env.getParameterValue(strName)
```

Parameter	Description
strName	The name of the parameter as supplied in the command file. For example, `script.js –param:document=c:\docs\myBqy –param:type=Query` In this case, strName is either document or type.

getParameterValues()

All values are obtained for a potentially multi-valued parameter as an array, based on the name. If the named parameter value does not exist, return null.

Example using getParameterValues():

```
var arrValues = env.getParameterValues(strName)
```

Parameter	Description
strName	The name of the parameter as supplied in the command file. For example, `script.js –param:document="[\"c:\\docs\\file1.bqy\", \"d:\\docs\\file2.bqy \"]"` In this case, strName is document.

getRepository()

Used to retrieve the repository artifact in whose context the script is running. If the script is running on the desktop, this is the file system.

Example using getRepository():

```
var repLocal = env.getRepository();
```

getScriptUuid()

The repository UUID of this script is retrieved.

Example using getScriptUuid():

```
var uuiScript = env.getScriptUuid();
```

Use this in EPM Workspace and on the desktop..

isDesktopMode()

Returns true if the script is running on the desktop.

Example using isDesktopMode():

```
var blnDesktop = env.isDesktopMode();
```

isServerMode()

Returns true if the script is running in EPM Workspace.

Example using isServerMode():

```
var blnWorkspace = env.isServerMode();
```

loadScript()

The JavaScript file is loaded and merged with the main script.

Example using loadScript():

```
env.loadScript(strPath, strDesc, uuiScript);
```

Example using loadScript():

```
env.loadScript(filScript, strDesc, uuiScript);
```

Parameter	Description
strPath	The string path that is used to locate the script file
filScript	The file object that references the JavaScript file
strDesc	**Optional:** The description that enables logging and debugging to identify whether an error occurred in the main script or a loaded script
uuiScript	**Optional:** The UUID of the script being loaded

Example 1: Use the string to search for the file in the same location as the script. If it fails to locate the file it searches the root of the script repository folder: in EPM Workspace the root is /Administration/Impact Manager/Script Repository, and on the desktop it is `C:\Hyperion\products\biplus\DDS\scripts`.

```
env.loadScript("lib_hysl_core.js");
```

Example 2: Use the file object (`env.loadScript(filScript, strDesc, uuiScript);`) to implement a similar mechanism to Example 1.

```
function _loadScript(in_strScript){
    var uuid, fil
    var folServer = "/Administration/Impact Manager/Script Repository/lib/"
    var folDesktop = "C:\\Hyperion\\products\\biplus\\DDS\\scripts\\"
    if (env.isServerMode()){
       uuid = cn_repLocal.getFileUuid(folServer + in_strScript)
    } else {
        uuid = folDesktop + in_strScript
    }
    fil = cn_repLocal.retrieveFile(uuid)
    env.loadScript(fil, in_strScript, uuid)
}
_loadScript("lib_hysl_core.js");
```

setParameterValues()

Used to set the value of named parameter. For example, if the same values are set up over and over, a separate script can be written that lists only a subset of the parameters, which are the only ones displayed in the parameter screen, setParameterValues() is used to set the others, and loadScript() is used to read in the original.

Example using setParameterValues()

```
setParameterValue(strName, strValue)
```

writeBqyDom()

A document is written to disk that is ready to import into the repository as a new version of an existing document or as a new document.

Example using writeBqyDom():

```
var filBqy = env.writeBqyDom(domBqy);
```

Parameter	Description
domBqy	The DOM that contains all the document nodes

Reporting and Analysis Repository: Repository Artifact

The repository artifact provides access to the features of the container that holds the documents. If the script is run in EPM Workspace it uses theOracle's Hyperion Reporting and Analysis repository. Publications that are stored here include; documents, connection information, and the folder hierarchy. If the script is run on the desktop, the file system represents the repository artifact with reduced features.

A repository artifact is created by calling `env.getRepository()`, and the object has these methods.

addVersion Method

A new version of the file is added to the repository. This method applies only in EPM Workspace. If the number and names of sections are not changed by your script, then the Interactive Reporting database connection (oce) information is not required.

Example 1 using addVersion:

```
var intV = objRep.addVersion(strUuid, objFile, strDesc);
```

Parameter	Description
strUuid	The document UUID
objFile	The file that contains the new content, created by calling env.writeBqyDom() method
strDesc	The description to add to the repository

Example 2 using addVersion:

```
var intV = objRep.addVersion(strUuid, objFile, strDesc, objOce);
```

Use this method format if the Interactive Reporting database connection files associated with the sections require change, or if you have modified the query or data model. This is used by Data Model Update; see “Using Impact Management Services” in the *Hyperion Workspace Administrator's Guide.*

Parameter	Description
strUuid	The document UUID
objFile	The file that contains the new content, created by calling the env.writeBqyDom() method
strDesc	The description to add to the repository
objOce	An object that represents section information for the Interactive Reporting document, including the Interactive Reporting database connection information that is associated with each query and data model.

Example 3 using addVersion:

```
var intV = objRep.addVersion(strUuid, objFile, strDesc, objOceOld,
objOceNew);
```

This form of the method is tailored to a specific situation, where the updated document has the same query and data model sections as the original (one for one mapping, the same number of each as the original, with identical names to the original) and you want to retain the settings of the previous version.

The oce details are copied from objOceOld (the previous version in repository) to objOceNew, as is.

JavaScript Update uses this form of the method, because it retains the oce settings of the previous version. See “Using Impact Management Services” in the *Hyperion Workspace Administrator's Guide.*

Parameter	Description
strUuid	The document UUID
objFile	The file that contains the new content, created by calling the env.writeBqyDom() method
strDesc	The description to add to the repository
objOceOld	An object that represents section information for the earlier version of the Interactive Reporting document, including Interactive Reporting database connection information that is associated with each query or data model.

convertBqyFileToUnicode Method

The Interactive Reporting document file is converted from a code page based format to Unicode format. To convert the format the desktop calls to Interactive Reporting Studio through its COM interface. This operation requires both `brioqry.exe` and `brioqry.tlb`.

Use this method if you are trying to convert an Interactive Reporting document to the latest format. All images are updated to the Resource Manager format as well as converting to Unicode.

Example using convertBqyFileToUnicode:

```
var objFile = objRep.convertBqyFileToUnicode(objFileOld, intCodePage)
```

Parameter	Description
objFileOld	The original file in the earlier format
inCodePage	The code page of the original document which is accessible from the DOM of the original file; that is, the attribute StdCodePage2

findFiles Method

A list of the files is retrieved that are contained within the folder represented by a UUID. The UUIDs of the files are returned in an array. The call can return the files in the folder, or all the files in a hierarchy of folders under that folder.

Example 1 using findFiles:

```
var clcFiles = objRep.findFiles(uuiFolder, uuiMimeType, blnRecursive)
```

Parameter	Description
uuiFolder	The UUID of the folder. On the desktop, this is the path name.
uuiMimeType	The file type to search for; for example, an Interactive Reporting document
blnRecusrive	False: examine just the folder or True: expand all sub-folders

Example 2 using findFiles:

```
var repLocal = env.getRepository()
var uuiFolder = env.getParameterValue("document")
if (repLocal.isFolder(uuiFolder){
    var uuiMime = env.getMimeTypeUuid("application/x-brioquery")
var clsUuid = objRep.findFiles(uuiFolder, uuiMime, true)
var a = 1
for (var it = clcUuid.iterator(); it.hasNext(); a++) {
    env.log("clcUuid[" + a + "] = " + it.next());
}
    env.expandRequestAction("document", clsUuid)
    return
}
```

folderExists Method

Returns true if the specified folder path exists within the repository.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using folderExists:

```
var blnExists = objRep.folderExists(strPath)
```

Parameter	Description
strPath	The path that represents the folder. In EPM Workspace, a folder is represented by a UUID, and on the desktop, the UUID is the same as a path.

getCurrentFolder Method

Returns the string that represents the current working folder.

Example using getCurrentFolder:

```
var uuiPath = objRep.getCurrentFolder()
```

Parameter	Description
uuiPath	The UUID of the folder path that is the current working folder

getFileUuid Method

The UUID that corresponds to the given absolute path of a file is returned. On the desktop, complete paths and the UUID are identical, but when writing scripts that are intended for the desktop or EPM Workspace, treat UUID values and paths as if they are different and make the extra calls that convert paths to UUID values.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using getFileUuid:

```
var uuiFile = objRep.getFileUuid(strPath)
```

Parameter	Description
strPath	The complete path that represents the file

getFolderContentsFor Method

A list of the names of the files are retrieved that are contained within the folder represented by the given path. The call returns the names in just the folder, or in the entire hierarchy of folders under the folder.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using getFolderContentsFor:

```
var arrNames = objRep.getFolderContentsFor(strPath, blnRecursive)
```

Parameter	Description
strPath	The complete path that represents the folder
blnRecursive	False: examine just the folder or True: expand all sub-folders

getFolderUuid Method

The UUID that corresponds to the given absolute path of a folder is returned. On the desktop, complete paths and the UUID are identical, but when writing scripts that are intended for the desktop or EPM Workspace, treat UUID values and paths as if they are different and make the extra calls that convert paths to UUID values.

Example using getFolderUuid:

```
var uuiFolder = repLocal.getFolderUuid(strPath)
```

Parameter	Description
strPath	The complete path that represents the file

getNameForUuid Method

The name that represents the leaf node of the path for the file referenced by the UUID is retrieved.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using getNameForUuid:

```
var strName = objRep.getNameForUuid(uuiPath)
```

Parameter	Description
uuiPath	The UUID of the files whose name is required

getPathForUuid Method

The path of the file referenced by the UUID is retrieved.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using getPathForUuid:

```
var strPath = objRep.getPathForUuid (uuiPath)
```

Parameter	Description
uuiPath	The UUID of the file whose path is required

getSubfolderPathsFor Method

A list of sub-folders for a folder is retrieved.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using getSubfolderPathsFor:

```
var arrPaths = objRep.getSubfolderPathsFor(uuiPath, blnRecusrive)
```

Parameter	Description
uuiPath	The UUID of the files whose name is required
blnRecursive	False: examine just the folder or True: expand all sub-folders

isFile Method

Returns true if the UUID is a file.

Example using isFile:

```
var bln = objRep.isFile(uuiPath)
```

Parameter	Description
uuiPath	The UUID of the object type that is being tested

isFolder Method

Returns true if the UUID is a folder.

Example using isFolder:

```
var bln = objRep.isFolder(uuiPath)
```

Parameter	Description
uuiPath	The UUID of the object type that is being tested

makeFolder Method

One or more folders are created.

Optional: Creates a hierarchy of folders if they do not exist.

This method is only available in EPM Workspace and does not apply to the desktop.

Example using makeFolder:

```
objRep.makeFolder(strPath, strDesc, blnRecursive)
```

Parameter	Description
strPath	The path that describes the folder to be created
strDesc	**Optional:** The description to be added to the repository for the path
blnRecursive	**Optional:** If this is set to true and the full parent sub-folders do not exist above the lowest node in the path, then folder creation starts at the first missing node and continues until all sub-folders in the path are created

publishBqyFile Method

An Interactive Reporting document is published or imported into the repository, configures the Interactive Reporting database connection mappings, and identifies how the EPM Workspace server treats sections. This function performs the work also done by the publishing wizard when an Interactive Reporting document is imported into EPM Workspace.

Example using publishBqyFile:

```
var uuid = objRep.publishBqyFile(objF, strN, strD, uuiF, blnD, strH, oceP)
```

Parameter	Description
objF	The Interactive Reporting document that is being published
strN	The file name being published
strD	The description associated with the file being published
uuiF	The folder UUID under which this is to be published
blnD	True indicates that the Interactive Reporting document contains dashboard sections
stwH	The section name that is displayed when the Interactive Reporting document is activated on the thin client and when 🏠 (Home) is clicked
oceP	An object that represents section information for the Interactive Reporting document, including the Interactive Reporting database connection information is associated with each query and data model

The example illustrates publishing a copy of an Interactive Reporting document. For example, if the selected file is called *sales analysis,* it is published with a name provided by the user, or if no name is provided as *Copy of sales analysis,* into the same folder as the source document. The Interactive Reporting database connection mappings from the source file are also copied to the new file so it can be processed in the same way as the source file. The script works on the desktop and in EPM Workspace.

Example: Publishing a copy of an Interactive Reporting document.

```
/**
 *
 * @param document Select the source to copy.
 * @inputType file_picker_single_value
 *
 * @param target Provide a name to call the copied file
 *
 */

var uuiSrc = env.getParameterValue("document");
var repLocal = env.getRepository();
var filSrc = repLocal.retrieveFile(uuiSrc);
var vrsSrc = repLocal.retrieveVersionedDocument(uuiSrc);
var strSrc = vrsSrc.getName();
var strTrg = env.getParameterValue("target");
if (strTrg == null){
   strTrg = "Copy of " + strSrc;
}
```

```
var uuiFolder = vrsSrc.getParentIdentity();
var domSrc = env.getDocument(filSrc, bqReadWriteDom, null);
var oceMapOld = vrsSrc.getSectionOCEMapping();
var oceMapNew = domSrc.sectionOCEPairInfos(uuiFolder);
for (var a = 0; a < oceMapOld.length; a++) {
    if (oceMapOld[a].isOCEEnabled()) {
        oceMapNew[a].setOCEDocument(oceMapOld[a].getOCEDocument());
        oceMapNew[a].setOCEEnabled(true);
    }
}
var strDesc = "this file was copied by a Rhino script from " + strSrc
var blnD =domSrc.isDashboard()
var strH = domSrc.getInitialTCSection()
repLocal.publish(filSrc, strTrg, strDesc, uuiFolder, blnD , strH,
oceMapNew);
```

retrieveFile Method

The latest or a specific version of a file is retrieved from the repository.

Example using retrieveFile:

```
var filBqy = objRep.retrieveFile(uuiBqy, intVersion)
```

Parameter	Description
uuiBqy	The UUID of the specified file. On the desktop the UUID is identical to the full path of the file.
intVersion	**Optional:** If omitted, then the latest version is obtained

retrieveVersionedDocument Method

The latest version of a versioned document object identified by UUID is retrieved. This method provides access to the document description, the keywords, the display name of the published file, the Interactive Reporting database connection file, and how the database connection maps to the document.

Note:

retrieveVersionedDocument is not for use on the desktop.

Example using retrieveVersionedDocument:

```
var vrsBqy = objRep.retrieveVersionedDocument(uuiBqy)
```

Parameter	Description
uuiBqy	The UUID of the specified document

The Node Object

An Interactive Reporting document is composed of hierarchically arranged node sets. Most nodes have commonalities and share methods and properties that apply to most Interactive Reporting document node types.

addChild()

An existing node is added as a child of another node.

Tip:

Useful to copy a node from one location to another.

Example using addChild():

```
nodTrg = nodMyNode.addChild(nodSrc)
```

Parameter	Description
nodSrc	References the source node to be replicated

addProperty()

An existing property is added to another node.

Tip:

Useful to copy a property from one node to another.

Example using addProperty():

```
prpRef = nodMyNode.addProperty(prpSrc)
```

Parameter	Description
prpSrc	References the source property to be replicated

cloneNode()

The entire node and its subordinates are cloned.

Example using cloneNode():

```
var bsndNew = nodMyNode.bsndSrc.cloneNode()
```

findNodeByPattern()

A single node, if any, that matches the specified pattern is retrieved.

Note:

If a node is not found, an exception is thrown.

Example using findNodeByPattern():

```
nodFound = nodMyNode.findNodeByPattern(strPattern)
```

Parameter	Description
strPattern	The search pattern

findNodesByPattern()

Nodes that match the specified pattern are retrieved.

Note:

No exception is thrown if none are found.

Example using findNodesByPattern():

```
arrNodes = nodMyNode.findNodesByPattern(strPattern)
```

Parameter	Description
strPattern	The search pattern

getChildren()

Returns an array of nodes that are directly under this node.

Example using getChildren():

```
arrNodes = nodMyNode.getChildren()
```

getNodeType()

Returns the type of the node; for example, a string.

Example using getNodeType():

```
strType = nodMyNode.getNodeType()
```

getPathWithContext()

Returns the path, as a slash-separated list of node names, with name attribute values that remove ambiguity as to the node identity.

Example using getPathWithContext():

```
strPath = nodMyNode.getPathWithContext()
```

getProperty()

The object that represents a given property is retrieved.

Tip:

Useful for getting values of out arrays.

Example using getProperty():

```
prpResult = nodMyNode.getProperty(strName)
arrValues = prpResult.getValues()
```

Property Object	Description
prpResult	An array of values that require reading or modification.

Parameter	Description
strName	The property name

getProperty() may be accompanied by a getValues() call; for example:

```
arrScripts = nodDocScripts.getProperty("EScript").getValues()
var startUp = arrScripts[0]
var shutDown = arrScripts[1]
var preProc = arrScripts[2]
var postProc = arrScripts[3]
```

Use the EScript property to access multi-valued properties that correspond to simple arrays; for example:

```
var someScript = docAnnotation.EScript[i]
```

hasProperty()

Returns true if the named property exists, otherwise returns false.

Example using hasProperty():

```
blnResult = nodMyNode.hasProperty(strName)
```

Parameter	Description
strName	The property name

Use hasProperty() rather than performing a Boolean test on the property name, as this returns false if the property is false or zero.

```
// this is not safe because if Offset is 0 it will return false
if (node.Offset){
    // do whatever is needed if the node has an Offset property
}

// this is safe
if (node.hasProperty("Offset"){
    // do whatever is needed if the node has an Offset property
}
```

removeChild()

The nominated child node is removed.

Example using removeChild():

```
nodMyNode.removeChild(nodChild)
```

Parameter	Description
nodChild	The child node to remove

removeProperties()

The properties identified by the array of names are removed.

Example using removeProperties():

```
nodMyNode.removeProperties(arrNames)
```

Parameter	Description
arrNames	The array of property names to delete

Note:

removeProperties() is useful to downgrade the format of an Interactive Reporting document to one that is generated by an earlier format or Interactive Reporting Studio Release. However, you are not required to do this because any properties or nodes not understood by Interactive Reporting Studio are ignored when the document is loaded, and therefore lost when the document is saved.

replaceChildNode()

The old child node is replaced with a new node.

Example using replaceChildNode():

```
nodRef = nodMyNode.replaceChildNode(nodChild, nodNew)
```

Parameter	Description
nodChild	A node that exists as a child

Parameter	Description
nodNew	The new node to replace the child

The BqyDocument Object

The BQY Document Object Model (DOM) provides access to the features of Interactive Reporting documents. Use this object to modify the internal properties of documents.

Example: Retrieving a DOM for a document.

```
var uuiBqySrc = env.getParameterValue("document");
var objRep = env.getRepository();
var bqySrc = objRep.retrieveFile(uuiBqySrc);
var domSrc = env.getBqyDocument(bqySrc, bqReadWriteDom, null);
```

Note:

A DOM is a collection of BQYNode objects arranged in a hierarchy or tree structure.

close()

The document is closed.

Example using close():

```
domSrc.close()
```

Note:

It is important to use this if a single script processes many documents as it saves resources.

compressBqy()

The Interactive Reporting document is compressed into the specified file.

Note:

Returns true if the file is compressed, or false if the file does not require compression.

Example using compressBqy():

```
var bln = domSrc.compressBqy(strNameOld, strNameNew, intInsHdrLen)
```

Parameter	Description
strNameOld	The Interactive Reporting document name to be compressed
strNameNew	The name of the compressed Interactive Reporting document. If it is identical to strNameOld, then the old uncompressed file is removed

Parameter	Description
intInsHdrLen	The number of bytes of Interactive Reporting Web Client (Insight) Header to skip to get to the main header

copy()

A copy is created on the specified section, and the copy is added to the DOM as a section node (Rpt.DocComp).

Example using copy():

```
var rdcTarget domSrc.copy(rdcSource)
```

Parameter	Description
rcdSource	The section to be copied, the Rpt.DocComp object

getInitialTCSection()

A string is returned that identifies the Home section of an Interactive Reporting document for publishing to the thin client.

Example using getInitialTCSection():

```
strName = domSrc.getInitialTCSection()
```

isBQYProcessable()

Determines whether the Interactive Reporting document contains at least one section to be processed.

Example using isBQYProcessable():

```
blnResult = domSrc.isBQYProcessable()
```

isCompressed()

Determines whether the document, from which the DOM derives, is compressed.

Note:

isCompressed() is useful if a requirement for the script is to change the compression status of an Interactive Reporting document.

Example using isCompressed():

```
blnResult = domSrc.isCompressed()
```

isDashboard()

Determines whether the Interactive Reporting document contains at least one dashboard section.

Example using isDashboard():

```
blnResult = domSrc.isDashboard()
```

sectionOCEPairInfos()

An array of Interactive Reporting database connection mappings is provided for the DOM.

Note:

These arrays are not the published Interactive Reporting database connection files associated with the document. However, you can associate the array of mappings with each query published Interactive Reporting database connection, and enable the document to access a data source defined in EPM Workspace.

Example using sectionOCEPairInfos():

```
oceMap = domSrc.sectionOCEPairInfos(uuiParentFolder)
```

Parameter	Description
uuiParentFolder	The folder UUID where the document is published

Example 1: Copying the Interactive Reporting database connection mappings from one DOM to another, and republishing the DOM as a new publication or a new version.

```
function copyBqy(in_repSrc, in_bqySrc, in_bqyTrg){
   var uuiFold = in_repTrg.getFolderUuid(in_bqyTrg.strFolder)
   var oceMapO = in_bqySrc.vrs.getSectionOCEMapping();
   var oceMapN = in_bqySrc.dom.sectionOCEPairInfos(uuiFold);
   for (var a = 0; a < oceMapO.length; a++) {
      if (oceMapO[a].isOCEEnabled()) {
         oceMapN[a].setOCEDocument(oceMapO[a].getOCEDocument());
         oceMapN[a].setOCEEnabled(true);
      }
   }
   var strD = "created by copyBqy"
   var blnD =in_bqyTrg.dom.isDashboard()
   var strH = in_bqyTrg.dom.getInitialTCSection()
   var uuiFound = hysl_getUuid(uuiFolder, in_bqyTrg.strName)
   var filBqy = in_bqyTrg.file
   var strN = in_bqyTrg.strName
   if (uuiFound != null){
      in_repSrc.addVersion(in_uuiToAdd, in_filBqy, in_strDesc)
   }else{
      in_repSrc.publishBqy(filBqy, strN, strD, uuiFold, blnD , strH,
oceMapN);
    }
}
```

Example 2: Publishing a new document and assigning a specific Interactive Reporting database connection to the queries of the new document.

```
var uuiFold = rep.getFolderUuid("/sales/monthly")
var oceMap = bqySrc.dom.sectionOCEPairInfos(uuiFold);
var uuiOCE = rep.getFileUuid("/OCE/salesInfo.oce")
for (var a = 0; a < oceMap.length; a++) {
    if (oceMap[a].isOCEEnabled()){
        oceMap[a].setOCEDocument(uuiOCE);
    }
}
var strD = "my description"
var blnD =bqySrc.dom.isDashboard()
var strH = bqySrc.dom.getInitialTCSection()
var filBqy = bqySrc.file
var strN = bqySrc.strName
in_repSrc.publishBqy(filBqy, strN, strD, uuiFold, blnD , strH, oceMap);
```

Method and Properties References

This topic includes reference tables for methods and properties.

Reference for env Methods

Method	Description
createTempFile()	Create a temporary file that is cleaned up when the script completes
expandRequestAction()	Add a new sub-task for each set of values
getBqyDocument()	Construct a DOM from the content of an Interactive Reporting document
getDescription()	Retrieve the description associated with the script
getFileLines()	Read the lines of a file and construct an array that contains one string per line
getLogLevel()	Retrieve the current default log level that is used when calling log()
getMimeTypeUuid()	Retrieve the UUID of the specified MIME type
getNullUuid()	Retrieve a null UUID constant
getParameterValue()	Retrieve the value of the specified script parameter
getParameterValues()	Retrieve all of the values assigned to a multi-value script parameter
getRepository()	Retrieve an object that can be used to access the content of the repository
isDesktopMode()	Determine whether the script is being run on the desktop
isServerMode()	Determine whether the script is being run in EPM Workspace
loadScript()	Load the content of another script into this script environment

Method	Description
log()	Post a message at the current default logging level
logAlways()	Post a message that is always written to the log
logClassName()	Post a message that contains the specified Java class name of the object
logDebug()	Post a message for debugging
logError()	Post a message associated with a detected error condition
logFatal()	Post a message associated with a detected error condition
logInfo()	Post an informational message
logWarn()	Post a warning message
md5Hash()	Generate an MD5 hash from the specified string
setLogLevel()	Set the default level at which logging is to be performed
setProgress()	Update the progress of the script
updateDescription()	Set a new description for this script invocation
writeBqyDom()	Write the specified DOM out to a file

Reference for Repository Methods

Method	Description
addVersion()	Add a version of a document
convertBqyFileToUnicode()	Convert the specified document from code page to Unicode
findFiles()	Find all files in a folder
getFileUuid()	Retrieve the UUID of the file with a specified path
getFolderUuid()	Retrieve the UUID of the folder with a specified path
isFile()	Determine whether the specified UUID represents a file
isFolder()	Determine whether the specified UUID represents a folder
publishBqyFile()	Import a file into the repository with the specified content
remapOCEs()	Remap the OCEs of the specified document to the provided set
retrieveFile()	Retrieve the document with the specified UUID as a temporary file
retrieveVersionedDocument()	Retrieve the versioned document associated with the specified UUID

EPM Workspace-Specific Repository Methods

Method	Description
changeToFolder()	Change the logical position within EPM Workspace to the specified folder path
folderExists()	Determine whether a folder with the specified path exists in EPM Workspace
getCurrentFolder()	Retrieve the path to the current folder where this script is located in EPM Workspace
getFolderContentsFor()	Retrieve the UUIDs of all files in the folder
getPathForUuid()	Get the path in EPM Workspace represented by the specified UUID
getSubfolderPathsFor()	Retrieve the UUIDs of all subfolders of the folder
makeFolder()	Create a subfolder with the specified name

Reference for Node Methods

Method	Description
addChild()	Add a child under this node
addProperty()	Add the specified property to this node
dump()	Dump the content of the node and the children of the node to standard output
findNodeByPattern()	Find one node that matches the specified pattern
findNodesByPattern()	Find all nodes that match the specified pattern
getChildren()	Retrieve a list of all the children of this node
getChildrenOfType()	Retrieve a list of all the children of this node with the specified node type
getContextualName()	Retrieve the logical name of this node
getNodeType()	Retrieve the type of this node
getPathWithContext()	Retrieve a string that represents the location of this node in the document, including contextual information to make the path unique
getProperties()	Retrieve a list of properties for this node
getProperty()	Retrieve the property of this node with the specified name
getRoot()	Retrieve the root node of the DOM in which this node is stored
hasProperty()	Determine whether this node has a property with the specified name
newNode()	Construct a node
removeChild()	Remove the specified child node

Method	Description
removeProperties()	Remove the specified list of properties from this node
replaceChildNode()	Replace the specified child node with the node provided
setChildren()	Replace the list of children of this node with the provided list

Reference for document

A document retrieved by using env.getBqyDocument() contains these properties.

Property	Description
DesignPassword	Password required to enter Design mode
DocumentPassword	Password required to open the document
EncryptedScripts	Determines whether scripts in the document are encrypted
EventScripts	Document-level scripts
Name	Document name
Path	Path to the document
Root_MyDocument	Root.MyDocument node
Root_MyResources	Root.MyResources node (or null if the document does not include Resource Manager data)
Sections	All sections contained in the document
Type	Retrieve the runtime class name
Unicode	Determines whether the document string content is in Unicode or code page format

The same document also contains these methods.

Method	Description
copy()	Copy the specified section to the document, rename it, if necessary, to avoid duplicates
getChartSections()	Retrieve a list of all chart sections
getChildrenWithRuntimeClass()	Retrieve all child nodes with a specified RuntimeClassName
getCodePage()	Retrieve the code page used by the document
getDashboardSections()	Retrieve a list of all the dashboard sections
getInitialTCSection()	Retrieve the home section identifier
getPivotSections()	Retrieve a list of all pivot sections

Method	Description
getQuerySections()	Retrieve a list of all query sections
getResultsSections()	Retrieve a list of all results sections
getSource()	Get the path to the Interactive Reporting document from which this document was loaded
getTableSections()	Retrieve a list of all table sections
isBQYPasswordProtected()	Determine whether the document has a password
isBQYProcessable()	Determine whether the document has at least one processable section
load()	Load a document from an Interactive Reporting document on disk
optimizeImages()	Optimize all of the Resource Manager images to remove duplicates
save()	Save the document to an Interactive Reporting document on disk
sectionOCEPairInfos()	Retrieve a list of all the document OCE mappings
setCodePage()	Set the document code page
setEndianness()	Set whether the document should be stored as big- or small-endian
setHeader()	Set the document header
setSource()	Set the path to the source from which this document was loaded

8

Dashboard Studio Inspector Utility

In This Chapter

Using the Dashboard Studio Inspector Utility

The Dashboard Studio Inspector Utility is a tool for prototyping your script that displays internal representations (node hierarchies and properties) of Interactive Reporting documents. By identifying the names of object properties, the utility enables you to write custom scripts that manipulate Interactive Reporting documents through the internal object model. These scripts are executed in EPM Workspace by the Impact Management Services Custom Update feature, or on the desktop through the Dashboard Development Services Update Utility or the Custom Update Script feature.

The user interface is organized in two panes; the left displays the node hierarchy, and the right displays the properties, values, and node data types.

To perform various actions (for example, to save a document or sort the node hierarchy), you can use the toolbar buttons (identified by their floating comments), commands provided from the menu bar, or keyboard shortcuts (identified next to the menu commands to which they correspond).

Opening One or More Documents

Multiple Interactive Reporting documents can be opened and inspected in multiple windows, and documents can be saved.

➤ To open one or more documents in the Inspector Utility, select **File**, then **Open**, or click 📂.

Note:

If a document is already open, a dialog box is displayed asking where to open the document; in this window or a new window.

Alternatively, if a document is open, you can drag another document onto the Inspector Utility. A dialog box is displayed asking where to open the document; in this window or a new window.

Searching for Values, Nodes, and Property Names

Use the Inspector Utility to search the internal representation of an Interactive Reporting document which contains a hierarchy of nodes and node attributes. Nodes can have no properties or subnodes, or multiple properties or subnodes. Properties are defined by a name, a type, and one or more values. The Interactive Reporting Studio property types are Byte (8 bit number), Word (16 bit number), DWord (32 bit number), Long (64 bit number), String (text), or Binary (binary content).

➤ To use the Inspector Utility to search for values, nodes, or property names:

1 **From the toolbar, select 🗔.**

The Search Results pane is displayed.

2 **Select Edit, then Find, or click 🔍.**

Find is displayed.

3 **From Search, select an option:**

- The entire document
- From this node

4 **From Search for values, select a parameter:**

- containing
- equal to
- starting with
- ending with

5 **Enter or select the value to which the parameter is applied.**

Searches are case-sensitive. You must enter or select a value, otherwise the search will not return any results.

6 **Optional: From Limit results to those in, enter or select either Nodes of type, or Properties named, or both.**

For example, in Nodes of type, enter `Box.Item` or in Properties named, enter `SizePY`.

7 **Click OK.**

The results are displayed in the Search Results pane. The status bar indicates how many matches are found.

Note:

When the Find dialog box is reopened, the default displays your last entered or selected value.

Tip:

You can enter any text into Search for values. For example, the pipe symbol (|), or a number such as 12.

If, in the Inspector Utility, you select multiple search items (for example, values, nodes, and property names), the results represent a combination of the results that would be returned by the individual search selections.

Searching from Selected Nodes

The Find From Here context menu item enables you to perform a search within a script. It calls `findNodesByPattern()` on the selected node and populates the Search Results pane with the nodes that are returned.

The most important parts of scripts are the searches that you use to locate DOM content that is relevant to what is intended.

To search in the DOM, you can use a search pattern that begins with “/” which is interpreted as an absolute path, from the root of the document. If you do not use “/”, the pattern is assumed to be relative to the node.

Search from a node rules:

- If a pattern begins with “/”, the search starts from the root of the DOM, otherwise from the selected node
- If square brackets (`[]`) are entered after a node name in the pattern, then the node must have a property that satisfies the given restriction
- A restriction can specify a property name, a value, or both
- More than one restriction for any node can be included
- Using “/” anywhere in the pattern, means there must be a node that matches what comes before “/”, with a child that matches what comes after “/”
- Using “//” anywhere in the pattern, means it is acceptable to have multiple intermediate nodes at that point in the path (If the pattern begins with “//”, it will match every node in the DOM that conforms to the rest of the pattern)

Note:

It is more efficient if you can determine an explicit pattern that matches the intended set of nodes, rather than using the root-based form of the pattern. This is also true for embedded “`//`”, if the base of the search is near the top of the tree hierarchy.

➤ To search from a node:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [1100 Custom].`

2 **Right-click the node, and select Find From Here.**

Find From Selected Node is displayed.

3 **Enter a search pattern.**

For example, enter `/BQY/Root.MyDocument/Rpt.DocComp[AnnotName=’1100 Custom’].`

4 **Click OK.**

The results are displayed in the Search Results pane (which opens automatically). The status bar indicates how many matches are found.

Generating Paths

When writing Impact Management Services scripts, you must locate the parts of the DOM that the scripts update. Using the Inspector Utility, you can generate and save paths to values, nodes, or property names.

➤ To use the Inspector Utility to generate paths to values, nodes, or property names:

1 **From the node hierarchy, select a node, and from the right pane, select the property name or value.**

The property name or value must be selected otherwise using the *Include* path-generation options in the JavaScript query string pane will not return results.

2 **Click (Show Search Path generation controls).**

JavaScript query string is displayed.

3 **Select a path-generation option:**

- Exact path to node
- Exact path to node and Include property name
- Exact path to node, Include property name, and Include property value
- All matching nodes—References nodes wherever they are located in the DOM and generates a wildcard search path to all nodes of one type (for example, to generate paths to both document scripts and dashboard shape scripts)
- All matching nodes and Include property name

- All matching nodes, Include property name, and Include property value

The text box displays the results of each selected option.

4 **Optional:** **To save the path, click Copy to clipboard.**

Alternatively, to save paths to nodes, right-click a node in the hierarchy, and select Copy to Clipboard, then Path (No Context) or Copy to Clipboard, then Path (With Context). See "Displaying and Copying Node Attributes" on page 175.

Note:

To investigate the DOM structure and for information on the accessing DOM properties, also see "Investigating the Impact Management Services DOM Structure" on page 143 and "Accessing Properties" on page 143.

Displaying and Copying Node Attributes

The Display context menu item enables you to display a node path with or without context, and the tree address.

➤ To display a node path:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [DataModel]`.

2 **Right-click the node, and select Display, then Path (No Context).**

Path is displayed with the following information; for example, `/BQY/Root.MyDocument/Rpt.DocComp`.

3 **To close the message box, click OK.**

➤ To display a node path with context:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [DataModel]`.

2 **Right-click the node, and select Display, then Path (With Context).**

Path With Context is displayed with the following information; for example, `/BQY/Root.MyDocument/Rpt.DocComp[RptName='DataModel']`.

3 **To close the message box, click OK.**

➤ To display a node tree address:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [DataModel]`.

2 **Right-click the node, and select Display, then Tree Address.**

Tree Address is displayed with the following information; for example, `[1, 0, 0, 23]`.

3 **To close the message box, click OK.**

Alternatively, the Copy To Clipboard context menu item enables you to copy and paste the information that is displayed under the Display context menu into a text editor; for example, Notepad.

➤ To copy and paste a node path:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [DataModel]`.

2 **Right-click the node, and select Copy To Clipboard, then Path (No Context).**

3 **Open a text editor; for example, Notepad.**

4 **Press Ctrl+V to paste the node path into Notepad.**

For example, `/BQY/Root.MyDocument/Rpt.DocComp` is pasted.

➤ To copy and paste a node path with context:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [DataModel]`.

2 **Right-click the node, and select Copy To Clipboard, then Path (With Context).**

3 **Open a text editor; for example, Notepad.**

4 **Press Ctrl+V to paste the node path into Notepad.**

For example, `/BQY/Root.MyDocument/Rpt.DocComp[RptName='DataModel']` is pasted.

➤ To copy and paste a node tree address:

1 **From the node hierarchy, select a node.**

For example, in `Sample_en_esm.bqy`, select `Rpt.DocComp [DataModel]`.

2 **Right-click the node, and select Copy To Clipboard, then Tree Address.**

3 **Open a text editor; for example, Notepad.**

4 **Press Ctrl+V to paste the tree address into Notepad.**

For example, `[1, 0, 0, 23]` is pasted.

Accessing the JavaScript Update Feature from the Dashboard Studio Inspector Utility

➤ To access the JavaScript Update feature from the Inspector Utility, select **File**, then **Open JavaScript Update**, or click .

Note:

This is equivalent to launching the Dashboard Development Services Update Utility. See Chapter 6, “Using the Dashboard Development Services Update Utility.”

Custom Update Script Feature Provided by the Dashboard Studio Inspector Utility

You can use the Custom Update Script feature provided by the Inspector Utility to locate, invoke, and perform generic transformations on Impact Management Services scripts. The parameters that you enter vary, depending on the requirements of the script to be updated (see “Entering Parameter Values” on page 178).

The feature is similar to the Custom Update feature in Impact Management Services; however it runs in the Dashboard Studio Inspector Utility.

Note:

If a script being called operates on an Interactive Reporting document under inspection, the Inspector Utility reloads the affected document when the script completes and the execution is terminated.

Performing Custom Updates

➤ To perform custom updates from the Inspector Utility:

1 **Select File, the Open Custom Update Script, or click .**

Choose Script is displayed.

2 **Click , and select a script.**

For example, from `C:\Hyperion\products\biplus\DDS\scripts\samples` in a standard installation, select `RevertImageResources.js`.

3 **Perform an action:**

a. Select **Specify parameters interactively**, and click **Next** to move to **Enter parameter values for script**, and enter parameters.

If the selected script contains no parameters, clicking Next directs you to Task options.

b. Select **Use a definition file**, and click 📂 (next to Select a Task Definition File), and select a file.

   Definition files are text files, all lines of which contain parameter values that are separated by commas.

4. **Click Next to move to Task options.**
5. **Select the Script Logging Level (see “Logging” on page 138).**
6. **Click Run.**

   Results of the Custom Update are displayed.

Entering Parameter Values

If, when performing a custom update, you selected to specify parameters interactively, you must, in the Enter parameter values for script screen, enter the preferred parameters.

Note:

The Enter parameter values for script screen varies, depending upon the selected script.

Note:

If you are required to specify a text file as part of entering parameters, the file must contain the relevant information for the customization.

Example: Using the RevertImageResources.js Script

Use the `RevertImageResources.js` script to undo merging of images and their relocation in an Interactive Reporting document, and to return it to a pre-9.3 release state.

➤ To use the `RevertImageResources.js` script in an Interactive Reporting document:

1. **Click 📂 (next to The document whose image resources are to be reverted), and select a Release 9.3 or later Interactive Reporting document.**
2. **Click Next to move to Task options.**
3. **Click Run.**

   Results of the Custom Update are displayed. A copy of the document is created with (images reverted) appended to the file name; for example, `Sample_en_esm (images reverted).bqy`.

Retrieving Information About Custom Update Errors

Custom update error messages are displayed in the Results pane of the Task options screen. Detailed error information is available in the `ddsInspector.log` file.

➤ To retrieve detailed error information:

1 **Open a text editor; for example Notepad.**

2 **Locate the logs folder, and open** `ddsInspector.log`**.**

In a standard installation, the logs folder is located under `C:\Hyperion\products\biplus\logs`.

Extending the Inspector Utility User Interface Using Scripts

Use Oracle's Hyperion® Impact Management Services scripts to extend the Inspector Utility user interface. Add features by writing JavaScript code and linking the code to the interface.

The Inspector Utility scripting connects to the user interface in two places: the toolbar and the context-sensitive menus (displayed when you right-click on items in one of the panes).

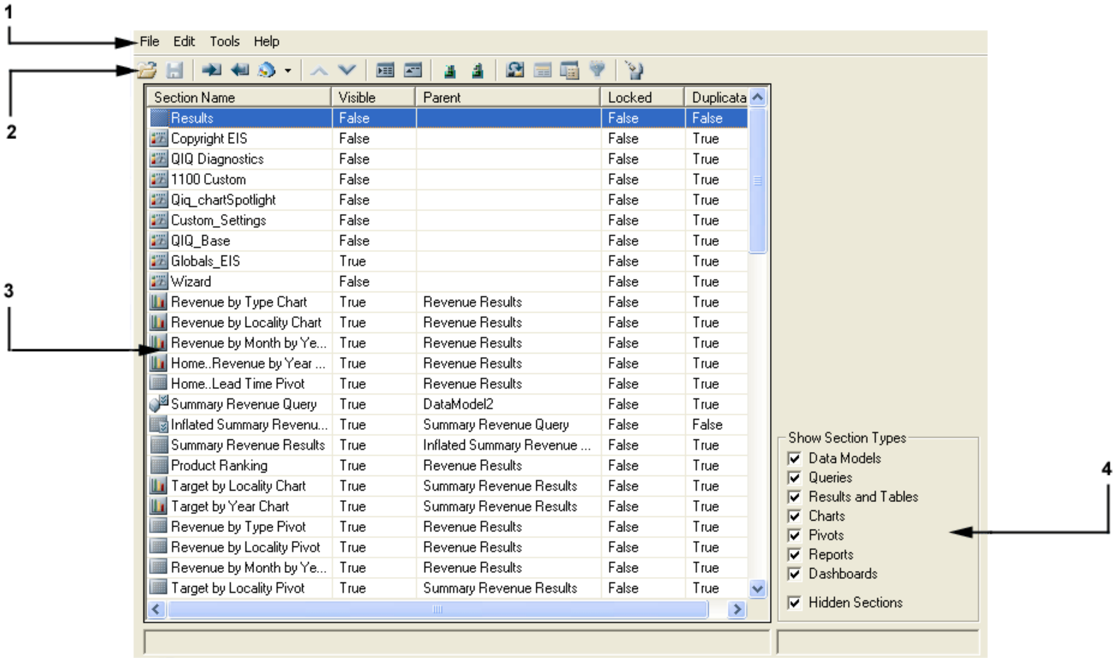
9

Dashboard Studio Optimize Utility

In This Chapter

Dashboard Studio Optimize Utility Interface Features

The Dashboard Studio Optimize Utility enables you to perform advanced editing options on Interactive Reporting documents. Using the utility, you can change the data source for a chart or pivot, rename a section, fix section references in code and section dependency lists, and delete multiple sections and their dependents.



Section Name	Visible	Parent	Locked	Duplicata
Results	False		False	False
Copyright EIS	False		False	True
QIQ Diagnostics	False		False	True
1100 Custom	False		False	True
Qiq_chartSpotlight	False		False	True
Custom_Settings	False		False	True
QIQ_Base	False		False	True
Globals_EIS	True		False	True
Wizard	False		False	True
Revenue by Type Chart	True	Revenue Results	False	True
Revenue by Locality Chart	True	Revenue Results	False	True
Revenue by Month by Ye...	True	Revenue Results	False	True
Home..Revenue by Year ...	True	Revenue Results	False	True
Home..Lead Time Pivot	True	Revenue Results	False	True
Summary Revenue Query	True	DataModel2	False	True
Inflated Summary Revenu...	True	Summary Revenue Query	False	False
Summary Revenue Results	True	Inflated Summary Revenue ...	False	True
Product Ranking	True	Revenue Results	False	True
Target by Locality Chart	True	Summary Revenue Results	False	True
Target by Year Chart	True	Summary Revenue Results	False	True
Revenue by Type Pivot	True	Revenue Results	False	True
Revenue by Locality Pivot	True	Revenue Results	False	True
Revenue by Month by Ye...	True	Revenue Results	False	True
Target by Locality Pivot	True	Summary Revenue Results	False	True

Show Section Types

☑ Data Models

☑ Queries

☑ Results and Tables

☑ Charts

☑ Pivots

☑ Reports

☑ Dashboards

☑ Hidden Sections

The four main interface areas of Dashboard Studio Optimize Utility are discussed in these topics:

1. Menu Bar
2. Toolbar
3. Section List

4. Filter Controls

Menu Bar

Table 10 provides the menu items and descriptions of the commands.

Table 10 Menu Bar Commands

Menu Item	Commands and Shortcuts (if available)	Description
File Menu	Open [Ctrl+O]	Open an Interactive Reporting document
	Close	Close an Interactive Reporting document
	Save [Ctrl+S]	Save an Interactive Reporting document
	Save As	Save an Interactive Reporting document with another name
	Exit [Ctrl+X]	Close the utility (If the Interactive Reporting document is modified, a prompt is displayed to save those changes before closing)
Edit Menu **Note:** The Edit menu is also available as a context menu for "Section List" on page 184	Move Section Up [Ctrl+U]	Move a section upward
	Move Section Down [Ctrl+D]	Move a section downward
	Show Sections [Ctrl+W]	Make a section visible (Visible sections are displayed in Interactive Reporting Studio Sections catalog)
	Hide Sections [Ctrl+H]	Hide a section (Hidden sections are not displayed in Interactive Reporting Studio Sections catalog)
	Rename Section [F2]	Rename a section and fix references to the original name
	Delete Sections [Del]	Delete selected and dependent sections
	Change Parent Section [Ctrl+P]	Change the parent of the selected sections so data is obtained from another section
	Change Report Dependency [Ctrl+G]	Change one section from which a report receives data

Menu Item	Commands and Shortcuts (if available)	Description
	Move Filters [Ctrl+L]	Move filters from one table or results to another
	Locked [Ctrl+K]	Lock a section
	Unlock [Ctrl+E]	Unlock a section
	Duplicatable [Ctrl+I]	Duplicate a section
	Unduplicatable [Ctrl+C]	A section is unable to be duplicated
	Pack Selected Sections [Shift+F6]	Compact the JavaScript in selected sections
	Pack All Sections [Shift+Ctrl+F6]	Compress the JavaScript in all sections
Tools Menu	Options	Display Options
	Consolidate images in Resource Manager	Remove duplicate images
Help Menu	Contents	Display the Help Table of Contents

Toolbar

Table 11 provides the toolbar buttons and descriptions.

Table 11 Toolbar Buttons

Button	Description
📂	Open an Interactive Reporting document
💾	Save an Interactive Reporting document
➡	Load the open Interactive Reporting document in Interactive Reporting Studio (It remains open in Interactive Reporting Studio for reference)
⬅	Load the open Interactive Reporting document in the utility into Interactive Reporting Studio (If it is open in Interactive Reporting Studio, it is closed without saving changes first. If the document is modified by the utility, a prompt asks for changes to be saved before it is closed)
🌐 ▾	Display Interactive Reporting Studio (Run the program or make it visible. Click the down arrow and select an Interactive Reporting document to open in the utility)
⌃	Move section up
⌄	Move section down

Button	Description
	Make sections visible
	Hide sections
	Lock sections
	Unlock sections
	Rename a section
	Change parent section
	Change report dependency
	Move filters
	Delete sections

Section List

The section list enables you to select one or more sections, and emulates standard Windows user interface behavior:

- Click a section to select it, and clear other sections
- Ctrl+click a section and click another section
- Shift+click a section and another section to select all sections in between
- Right-click the section list to display the Edit menu as a context menu

Filter Controls

Use the section type filter check boxes to selectively show or hide section types, or to show or hide sections based on the visible attribute.

Other Dashboard Studio Optimize Utility Features

Other features that are not visible on the Dashboard Studio Optimize Utility interface, include the Options dialog box, the Consolidate Images in Resource Manager feature, and the utility Unicode functionality.

Options Dialog Box

To access the Options dialog box, select Tools, then Options.

Options contains three functions:

- **Back-up BQYs before opening them**—Select whether the utility performs a backup of Interactive Reporting documents before opening them, and if so, where to save them (Backups of Interactive Reporting documents are saved in a specified folder, with a date-and-time stamp appended to the name. Interactive Reporting document backups are never deleted by the utility)
- **Language**—Enables the language option of the Dashboard Studio Optimize Utility to be set (A necessary option for localized or Unicode releases, as different languages display characters differently)
- **Set Font**—Launches a Font dialog box to enable the font style to be configured for the Dashboard Studio Optimize Utility (A necessary option for localized or Unicode releases, as different fonts display characters differently. Users must select an appropriately sized font that suits the UI, otherwise characters may be too large and cut off)

Consolidate Images in Resource Manager

Dashboard Studio Optimize Utility enables you to consolidate duplicate images into the Resource Manager. All instances of an image are changed to reference the single copy of the image in the Resource Manager. The Interactive Reporting document file size and memory footprint are reduced, which improves the loading speed, and makes reuse of existing images from the Resource Manager possible when creating dashboards and reports. When complete, a summary message box is displayed.

To merge duplicate images, select Tools, then Consolidate images in Resource Manager.

See “Resource Manager” in the *Hyperion Interactive Reporting Studio User's Guide.*

Unicode Functionality

The Dashboard Studio Optimize Utility automatically converts old code page based Interactive Reporting documents to Unicode before use, enabling documents with different languages to be optimized.

Editing Interactive Reporting Documents with Dashboard Studio Optimize Utility

This topic explains the operations you can perform on Interactive Reporting documents with the utility.

Move Sections Up or Down Feature

The move section up or down feature sets the order that the sections are displayed in the Interactive Reporting document, and the processing order of the queries. The function is available only when all sections are visible. For example, when all filters and one section are selected.

Show or Hide Sections Feature

The feature enables you to show or hide selected sections. Interactive Reporting Studio enables you to make multiple hidden sections visible simultaneously, by selecting Edit, then Unhide Section, but does not enable you to hide multiple sections in one operation.

Lock and Unlock Sections Feature

Use the feature to lock or unlock sections in an Interactive Reporting document. A locked section cannot be modified except when using Interactive Reporting Studio, which enables administrators to make sections of an Interactive Reporting document read-only.

Lock and unlock works with the duplicatable and unduplicatable feature. A section that is unlocked is duplicatable. A section that is locked can be duplicatable or unduplicatable.

The feature is provided to prevent end users from being able to change dashboard sections. For example, an end user who uses a plug-in, such as Interactive Reporting Web Client, is prevented from making changes to sections if the lock flag is activated.

Duplicatable and Unduplicatable Sections Feature

Use these options to duplicate a section or flag a section as unduplicatable. A section flagged as Duplicatable can be duplicated, and if the original section is locked the duplicate section is created as unlocked. A section flagged as Unduplicatable cannot be duplicated. The duplicatable and unduplicatable flags are available for end users who use a plug-in, such as Interactive Reporting Web Client.

Note:

Results are unduplicatable, so they are displayed in the utility as Locked–False and Duplicatable–False. This option takes precedence over the application of duplicatable or unduplicatable flags on other sections.

Rename Sections Feature

Interactive Reporting Studio enables you to rename sections, but it can cause problems with reports or other sections, and the JavaScript code that references the sections by name.

- If a section used in report tables is renamed, the report displays `Script(x):uncaught exception`

- If an embedded section, that is set to Hyperlink is renamed, the code behind the embedded control no longer works because it still references the original name (If the JavaScript references sections by name, the code displays an identical problem)

When the utility renames a section it repairs all references to the original name, including references from report tables and in JavaScript of the form "*original-section-name*", which must include the quotation marks.

Processing of scripts catches most code that references the section. The utility also fixes references to the original section name in dashboards created with Dashboard Studio, including embedded target information and navigation controls.

Note:

Filter Aliases are recognized in the Optimize Utility when a section is renamed.

Delete Sections Feature

The feature enables you to delete one or more sections, and sections dependent on deleted sections are also removed.

The Delete Sections dialog box is a warning that is displayed when a delete sections operation is initiated upon sections with dependents. Sections that cannot work without the selected sections are listed for deletion in the dialog box. Report or other sections that use the selected sections (or their dependents) are listed at the bottom of the dialog box. These sections do not stop working but do contain errors.

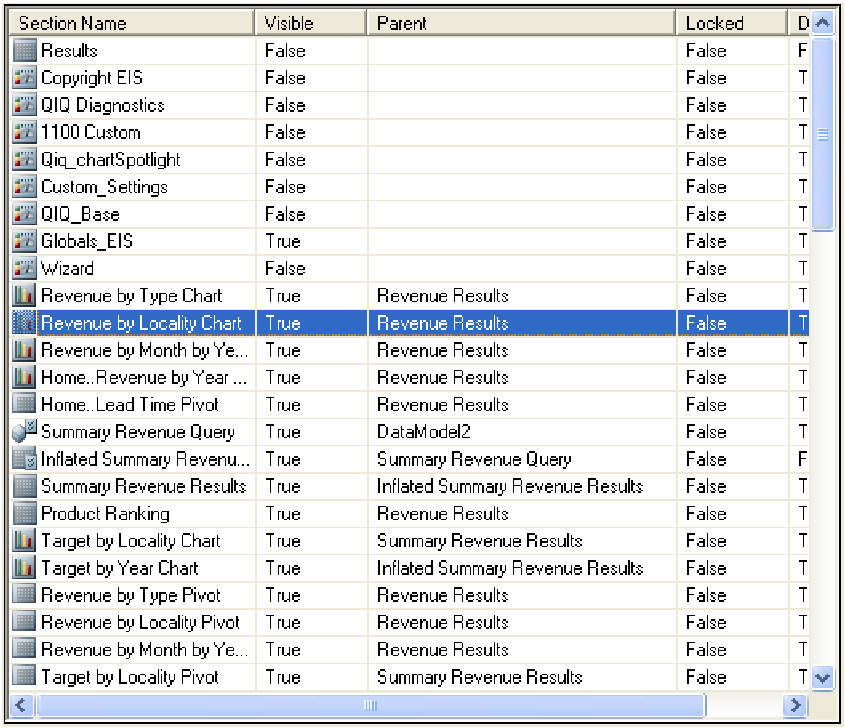
Using the Change Parent Section Feature

The feature enables you to change the parent of a section so it obtains data from another section. The most common use for the operation is to increase the speed of dashboards that rely on results with many or complex computed items and filters. By creating a table from the results, and placing the filters on the table, recalculating the computed items is avoided each time the filters are set.

When the operation is performed, all charts and pivots that depend on the results must be re-created in the table.

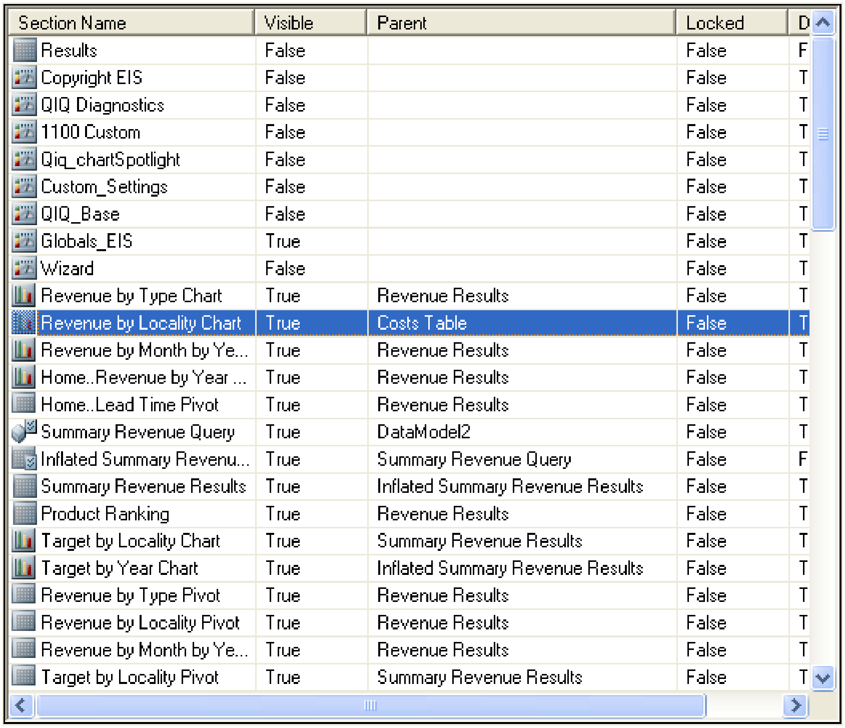
Figure 7 and Figure 8 illustrate the effect of the change parent section operation. Figure 7, displays a chart as dependent on results. The utility enables you to change the chart parent section to a table. Figure 8 displays the chart dependency changed to a table.

Figure 7 Chart Dependent on Results



Section Name	Visible	Parent	Locked	D
Results	False		False	F
Copyright EIS	False		False	T
QIQ Diagnostics	False		False	T
1100 Custom	False		False	T
Qiq_chartSpotlight	False		False	T
Custom_Settings	False		False	T
QIQ_Base	False		False	T
Globals_EIS	True		False	T
Wizard	False		False	T
Revenue by Type Chart	True	Revenue Results	False	T
Revenue by Locality Chart	True	Revenue Results	False	T
Revenue by Month by Ye...	True	Revenue Results	False	T
Home..Revenue by Year ...	True	Revenue Results	False	T
Home..Lead Time Pivot	True	Revenue Results	False	T
Summary Revenue Query	True	DataModel2	False	T
Inflated Summary Revenu...	True	Summary Revenue Query	False	F
Summary Revenue Results	True	Inflated Summary Revenue Results	False	T
Product Ranking	True	Revenue Results	False	T
Target by Locality Chart	True	Summary Revenue Results	False	T
Target by Year Chart	True	Inflated Summary Revenue Results	False	T
Revenue by Type Pivot	True	Revenue Results	False	T
Revenue by Locality Pivot	True	Revenue Results	False	T
Revenue by Month by Ye...	True	Revenue Results	False	T
Target by Locality Pivot	True	Summary Revenue Results	False	T

Figure 8 Chart Dependent on Table



Section Name	Visible	Parent	Locked	D
Results	False		False	F
Copyright EIS	False		False	T
QIQ Diagnostics	False		False	T
1100 Custom	False		False	T
Qiq_chartSpotlight	False		False	T
Custom_Settings	False		False	T
QIQ_Base	False		False	T
Globals_EIS	True		False	T
Wizard	False		False	T
Revenue by Type Chart	True	Revenue Results	False	T
Revenue by Locality Chart	True	Costs Table	False	T
Revenue by Month by Ye...	True	Revenue Results	False	T
Home..Revenue by Year ...	True	Revenue Results	False	T
Home..Lead Time Pivot	True	Revenue Results	False	T
Summary Revenue Query	True	DataModel2	False	T
Inflated Summary Revenu...	True	Summary Revenue Query	False	F
Summary Revenue Results	True	Inflated Summary Revenue Results	False	T
Product Ranking	True	Revenue Results	False	T
Target by Locality Chart	True	Summary Revenue Results	False	T
Target by Year Chart	True	Inflated Summary Revenue Results	False	T
Revenue by Type Pivot	True	Revenue Results	False	T
Revenue by Locality Pivot	True	Revenue Results	False	T
Revenue by Month by Ye...	True	Revenue Results	False	T
Target by Locality Pivot	True	Summary Revenue Results	False	T

➤ To use the Change Parent Section feature:

1. From the section list, select one or more sections, and click .

   Select New Parent is displayed.

2. Select another parent section, and click **OK.**

   If the parent of one or more charts or pivots is changed, the utility checks to see if those sections are used on reports, and prompts to fix them.

Note:

The Optimize Utility can re-parent tables.

Using the Change Report Dependency Feature

Unlike charts and pivots, reports may depend on multiple sections.

The change report dependency feature is similar to a change parent section operation, but the report dependencies are selected to change.

➤ To use the Change Report Dependency feature:

1. From the section list, select one or more sections, and click .

   Change Report Dependency is displayed.

2. From **Change references from section**, select a section to be used by the report.
3. From **To section**, select a replacement section.
4. Click **OK** to apply changes.

Note:

Reports can contain BLOB images from a database or aggregate tables. For information on BLOB image support, see the *Hyperion Interactive Reporting Studio User's Guide.*

Moving Filters Feature

The Move Filters feature enables you to move one or more filters from one section to another. The feature is enabled only when one table or result set that contains filters is selected.

➤ To move a set of filters:

1. Select the table or result set from which to move the filters, and select **Edit**, then **Move Filters**, or click .

   Move Filters is displayed.

Move Filters only lists supported filters that do not exist in the destination table or results. Supported filters contain a topic item that shares a name with the filter.

2 **From Select Filters to move, select the filters.**

3 **From Move Filters to Section, select the table or results to move filters to.**

If no filters can be moved to the selected table or results, a message is displayed at the bottom of Move Filters.

4 **Click OK.**

Note:

The Filter Enhancement feature (see "Using the Filter Enhancements Properties Frames" on page 86) means that ESM and runtime documents are incompatible with the Move Filters feature of the Optimize Utility in Release 9.3 and earlier.

Packing JavaScript Feature

The Pack selected sections and Pack all sections features, compress the JavaScript in sections by removing all leading and trailing white-space (spaces and tabs), blank lines, comments, and trace statements.

➤ To pack JavaScript:

1 **Select Edit, then Pack selected sections or Edit, then Pack all sections.**

Pack JavaScript is displayed.

2 **Select Remove Dashboard Studio trace statements, to remove the `Qiq_enter`, `Qiq_trace`, and `Qiq_exit` statements.**

3 **Click OK.**

A status bar indicates that the sections are being packed.

4 **Click Load this BQY into Interactive Reporting Studio to observe that the packed sections perform equally to the uncompressed sections.**

The Interactive Reporting document is now smaller.

Part II

Configuration and Customization Files

In Configuration and Customization Files:

10

Using the Annotation System

In This Chapter

About Annotations

Annotations are text labels or other graphic objects which are toggled on or off when ? is clicked to activate Help. Annotations can be used to highlight features of a dashboard which require additional explanation.

Creating Annotations

This procedure describes how to create and view annotations.

➤ To create and test annotations:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **From the catalog pane, expand Graphics.**

3 **Drag a text label onto the frame, right-click the label, and select Properties.**

Properties is displayed.

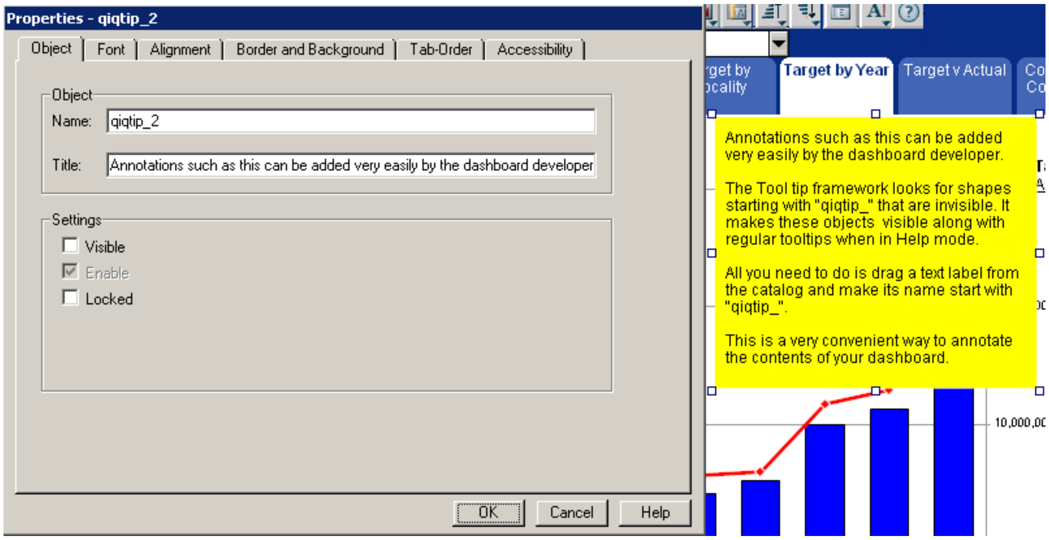
4 **In Name, enter** `qiqtip_`*`<name>`***.**

The name must begin with `qiqtip_`. For example, enter `qiqtip_Limits`.

5 **In Title, enter the annotation.**

6 **Optional: Enter the annotation directly into the text label.**

7 **Clear Visible, and click OK.**



Object

Name: qiqtip_2

Title: Annotations such as this can be added very easily by the dashboard developer

Settings

☐ Visible

☑ Enable

☐ Locked

Annotations such as this can be added very easily by the dashboard developer.

The Tool tip framework looks for shapes starting with "qiqtip_" that are invisible. It makes these objects visible along with regular tooltips when in Help mode.

All you need to do is drag a text label from the catalog and make its name start with "qiqtip_".

This is a very convenient way to annotate the contents of your dashboard.

8 **Repeat steps 3–7 to insert other annotations.**

9 **Format the annotation.**

10 **Press Ctrl+D to exit Design mode, and click ?, on the dashboard top panel.**

The annotation is visible on the frame.

The position of the annotation is important. Text labels are visible if positioned in front of a view-only chart. However, text labels are typically situated behind pivots, drop-down lists, check boxes, and other active objects, so it is recommended that annotations are not positioned on top of these objects. This is a standard feature in Interactive Reporting Studio.

If a text label is positioned on top of a graphic that contains JavaScript, the graphic cannot be clicked and the code cannot be executed, even if the text label is not visible. This is a standard feature in Interactive Reporting Studio.

11 Using Auto Z

In This Chapter

Default Auto Z Column Feature

A threshold is associated with Auto Z, so that the default Auto Z column is present only in the pivot column label or chart z-axis (depth) if the number of available values is less than or equal to the threshold specified in "Step 7: Configure Properties" on page 75. If more values are present than the specified threshold, the column is automatically removed, preventing the section from having excessive values, and causing it to be unreadable.

Creating Auto Z Columns

You can create an Auto Z column that is displayed for charts and pivots when the threshold is exceeded.

➤ To create the default Auto Z column:

1 **In Interactive Reporting Studio, navigate to the table or results, right-click, and select Add Computed Item.**

Computed Item is displayed.

2 **In Name, enter `_autoz<filter name>`.**

For example, to create a default column for the Quarter filter, create a computed item called *_autozQuarter*. The selected filter must be the first item in the Filters drop-down list. The filter name specified in the default column must match the name of the column in the table or results. Do not replace spaces with underscores.

3 **In Definition, enter the text to be displayed on the column.**

For example, enter `'All Quarter Values'`.

4 **Click OK.**

5 **Select the pivot to be associated with the Auto Z feature.**

6 From the catalog pane, drag the computed item column onto Data Layout and into Column Labels.

If the computed item cannot be dragged onto Data Layout, ensure that Pivot, then Refresh Data, then Manually is *not* selected. If it is selected, select another option and then drag the computed item into Column Labels. Reset Pivot, then Refresh Data, then Manually.

7 In **Step 4: Associate Frames with Charts, Pivots, or Tables** of Dashboard Studio, select **Allow first filter to be added to Z/Top Axis.**

8 Click to associate the sections with the frames.

9 Click ✓ to apply the selection to the document.

10 In **Step 6: Configure Filters** of Dashboard Studio, ensure that the filter is the first item in Active Filters, and click ✓ to apply the filters.

11 **Optional: In Step 7: Configure Properties** of Dashboard Studio, enter a value for the Auto Z/Top Axis Threshold to restrict the pivot to the valid number of columns, and click ✓.

The default threshold value is 4.

The Auto Z column is displayed in the configured section if the threshold is exceeded.

Checklist for Auto Z Creation

1. The column must be a local filter.
2. In Step 4 of Dashboard Studio, you must select Allow first filter to be added to Z/Top Axis for the frame.
3. In Step 6 of Dashboard Studio, you must reorder the filter to be the first filter for the frame.
4. In Step 7 of Dashboard Studio, you must set a valid threshold, to restrict the pivot to only the valid number of columns.

12 Spotlighting Charts

In This Chapter

Configuring for Chart Spotlighting

Interactive Reporting Studio enables you to color code cells in pivots and tables, based on a condition. Dashboard Studio extends the functionality by enabling you to color code the bars of vertical and horizontal charts, the segments of pie charts, or shapes, such as rectangles (see the Dashboard Studio Maps Component shapes information under "Configuring the Map Blank Frame" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*).

Dashboard Studio templates include the logic to examine the legend of charts, and to configure the color properties of each legend item according to the calculated results in a computed column. The logic is called whenever a frame is activated or a filter is changed on a frame that contains a spotlighted chart.

A reservation function is implemented for spotlighted charts. When a chart is spotlighted, it is reserved, and cannot be influenced by features that may cause it to lose the applied colors. Influencing features include switching between two-dimensional and three-dimensional charts, switching the legend control, changing chart types, applying the Quick Slice feature, applying chart shading, and so on.

A spotlight example is a chart that identifies bands of performance level for each locality according to a business rule, which would have these expected results:

```
Total Amount for a Locality < 1000, then bar = Red
Total Amount for a Locality < 5000, then bar = Light Orange
Total Amount for a Locality < 10000, then bar = Green
Total Amount for a Locality >= 10000, then bar = Aqua
```

To spotlight charts based on other conditions or elements, replace the columns and variables.

These procedures assume that you are working on a dashboard created with Dashboard Studio, and that the necessary Interactive Reporting Studio sections are also created.

Also see "Using the Color Palette Properties Frame" on page 98.

Creating Summary Tables

The Create Summary Table frame is designed to automate the process of building summary tables by creating columns to group by, creating computed items, sorting columns, local tables, and filters. Using the Create Summary Table frame saves development time.

Create Summary Table creates a structure that contains the same number of rows as there are bars or segments in a spotlighted chart, or shapes in a spotlighted shape set; for example, a map. See "Dashboard Studio Maps Component" in the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*).

➤ To use the Create Summary Table frame:

1 **In Interactive Reporting Studio, navigate to Create Summary Table.**

To access the configuration frame, use Sections or the Configuration drop-down list found in Dashboard Studio.

2 **In Grouping Column Configuration, select a Table/Results Section that contains your data.**

3 **Optional: To view the selected section, click ▦.**

4 **Optional: Enter a Grouping Column Name to override the default created by the system.**

If you are grouping multiple columns, enter a Grouping Column Name. For example, grouping country and city, where city is not a unique entity, the user can override the default name which is the concatenation of the column names; for example, Country_City. The computed dimension is a concatenation of the column values.

5 **Select an available column to highlight in the chart.**

For example, if you are using `Sample_en_esm.bqy`, select State.

6 **Sort columns by moving selected columns between Available Columns and Grouping Columns by clicking > and <.**

If Grouping Columns contains multiple items, create a computed item which is a concatenation of the selected columns, otherwise the selected column is the grouping column. Add the grouping column to the Interactive Reporting Studio Sort line of the section.

7 **Optional: Reorder Grouping Columns by selecting a column and clicking ^ or v.**

8 **In Summary Table Configuration, enter a Summary Table Name.**

For example, enter `State_Color`. (This name must match the shape set name used later in the spotlighting process).

9 **Select a Function to create an aggregated total for each label value.**

For example, select Sum.

The function is the summary formula used when creating Summary Fact Columns. The function value is combined with selected Available Columns when > is clicked to generate Summary Fact Columns.

10 **Move selected columns between Available Columns and Summary Fact Columns by clicking > and <, to identify the columns for which to create summary computed items.**

For example, select Revenue to create a summary computed item called *Revenue_._Sum_._*.

11 **Optional: Reorder Summary Fact Columns by selecting a column and clicking ∧ or ∨.**

12 **Optional: To add more columns, move selected columns between Available Columns and Summary Table Columns by clicking > and <, to identify which columns are added to the summary table.**

13 **Optional: Reorder Summary Table Columns by selecting a column and clicking ∧ or ∨.**

14 **Perform an action:**

- To remove all settings, click Clear
- To build the summary table, click Create

All validation is performed. The summary table is created.

The computed column that determines the color to use for a bar or a shape is typically located in this section. Use the lower section of the Create Summary Table frame to create the computed column, or if the column is complex, create it directly in the table and only reference it through the Create Summary Table frame.

Proceed to Building and Dragging Charts onto a Frame.

Building and Dragging Charts onto a Frame

Create the chart to be spotlighted and move it onto a frame. The chart must be a vertical bar, horizontal bar, or pie, and cannot contain z-axis (depth) values. Other chart types cannot be spotlighted.

➤ To build and drag chart onto a frame:

1 **Use the Interactive Reporting document from the Creating Summary Tables procedure.**

2 **From the *State_Color* table, create a chart and name it.**

Design mode is enabled.

3 **Drag the column (for example, State) to the x-axis, and drag the column to use as a comparison (for example, Revenue_Sum) to Fact (y-axis).**

4 **Create a frame to hold the chart.**

For example, name the frame *State_Revenue.*

5. Drag the chart onto the frame and format it.
6. Press **Ctrl+D** to exit Design mode.
7. Save the document.

Note:

Charts that include a z-axis (depth) value cannot be spotlighted.

Tip:

Ensure that the legend for the chart is set to the x-axis.

Proceed to Associating Charts.

Associating Charts

Use Dashboard Studio to associate the spotlighted charts and frames.

➤ To associate a chart:

1. In Dashboard Studio, navigate to Step 4: Associate Frames with Charts, Pivots, or Tables.
2. Click to associate frame with the chart.

   The code can now locate the business rules and apply them to the bar colors on the chart. As filters are changed, the colors are recomputed to reflect the performance of each filter; for example *state*.

   Proceed to Spotlighting the Chart.

Spotlighting the Chart

Use the Chart Spotlight Properties frame which includes the optional creation of a computed item using an expression builder.

➤ To use the Chart Spotlight Properties frame:

1. Use the Interactive Reporting document from the Building and Dragging Charts onto a Frame procedure.
2. From **Sections**, or the Configuration drop-down list found in Dashboard Studio, select Chart Spotlight Properties.
3. From **Dashboards with Charts**, select an option:
   - All Dashboards—Sections in a dashboard with an associated vertical bar, horizontal bar, or pie chart
   - Configured Dashboards—Contains at least one Chart Spotlight specification

- Unconfigured Dashboards—Contains no Chart Spotlight specifications

These options filter the frames that are displayed in the drop-down list. That is, only vertical bar, horizontal bar, or pie charts, without z-axis (depth) values are listed.

4 **Select a section from the drop-down list.**

For example, select State_Revenue.

5 **Optional: To view the selected section, click .**

If no section is selected, is not selectable.

6 **Under Spotlight Configuration, select an item from Chart Section/Shape Set.**

For example, select the chart from the Building and Dragging Charts onto a Frame procedure. The drop-down list displays names for charts on the selected dashboard and sets of shapes that can be colored by the spotlight feature. Shape examples include circles or rectangles that represent color indicators or points on a map, and so on. You must select the shape set name that matches the summary table name. Shape set names cannot contain spaces.

Table Section is displayed. For example, State_Color is displayed.

7 **From Table Section, to view the table, click (next to the table text label).**

The displayed table is the source for the selected chart.

If no section is selected, is not selectable.

8 **From Labels Column, select a column that associates with the x-axis of the selected chart or one of the shapes.**

For example, select State.

The remaining tools on Chart Spotlight Properties assist with defining computed items. Proceed to Defining Color Computed Items.

Defining Color Computed Items

There are two ways to define color computed items; externally outside the Chart Spotlight Properties frame or within the Chart Spotlight Properties frame.

Using an Externally Defined Column

You can create and maintain the column that determines bar or shape colors externally to the Chart Spotlight Properties frame. In this case, use the Chart Spotlight Properties frame drop-down list to select the existing column and click Apply.

Using a Column Defined in the Chart Spotlight Properties Frame

The created or modified computed item is defined using the tools that remain on the Chart Spotlight Properties frame. Computed items cannot be read from the object model, so settings are remembered and loaded as part of the specification. If the user modifies the specification directly, the modification is unable to be read.

➤ To define the computed item:

1 **On Chart Spotlight Properties, perform an action:**

a. Ensure **Create Color Column** is selected, and enter the name of the computed column to create in the text box.

For example, enter `State_Color`.

b. Do not select **Create Color Column**.

The drop-down list displays the computed item to be modified. Computed and non-computed columns are displayed, separated by a blank line. Computed columns are at the top of the list.

2 **From Value Column, select a column that contains the values that determine which colors to apply.**

3 **Specify the color condition by selecting an Operator and Value.**

a. Select an **Operator**.

For example, select > (greater than). Operator lists a set of logical operators to be used in the expression that is being generated.

b. In **Value**, enter characters (numbers or letters) to use in expressions.

For example, enter `55000`.

4 **From the drop-down list, select a palette to change the color scheme of the displayed option buttons.**

For example, select Default 4. If no palettes are displayed use the Color Palette Properties frame to configure a palette. See “Using the Color Palette Properties Frame” on page 98.

5 **Select an option (color) to combine with Operator and Value to create an expression.**

6 **Move selected specifications, by clicking › and ‹.**

The specification is displayed as it is used to construct the computed item.

7 **Optional: To enable Selected Condition, select a specification.**

The color associated with the expression is displayed.

8 **Optional: Reorder selected specifications, by selecting a specification and clicking ˄ or ˅.**

9 **Perform an action:**

- To discard specifications and to reload the original settings, click Restore (This occurs at document Startup)

- To remove the specification for a chart or shape set, click Clear
- To remove all current specifications, click Clear All
- To load spotlight settings, click Apply (Clicking Apply after clicking Clear All also results in the specification being cleared)

10 From Dashboards with Charts, click , to verify that the chart is spotlighted correctly.

13 Template Structure

In This Chapter

Frame Types

A Dashboard Studio template is composed of GUI frames and sections that contain the JavaScript infrastructure. Dashboard Studio reads the template description and constructs the interface required to build the dashboard that the template is designed to deliver. As a result, Dashboard Studio can provide various steps or elements, depending on the template used.

A dashboard created with Dashboard Studio is comprised of frames (dashboard sections that contain the FrameType property) and other added sections. The FrameType property is a text label called *FrameType*. FrameTypes determine the functions of sections.

Tip:

The FrameTypes outlined in this chapter are hidden in Interactive Reporting Studio, and can be accessed by selecting View, then Unhide Section.

Wizard

FrameType = 1000

The key difference between ESM and runtime versions of dashboards is that an ESM contains a section called *Wizard*. The Wizard section is the interface between the template and Dashboard Studio. The Wizard stores the knowledge about Interactive Reporting Studio and the structure of Interactive Reporting documents and templates. Without this section, Dashboard Studio has no way of knowing how to deal with the Interactive Reporting document. Runtime versions do not contain this section and are not connected to Dashboard Studio.

Special functions exist within the Wizard. These functions are called by Dashboard Studio as the building process moves from step to step. Dashboard Studio provides *extract* and *update* functions for each step. Extract functions gather the current property settings at the start of each step and make them available to Dashboard Studio. Update functions write the settings made in Dashboard Studio to the Interactive Reporting document at the end of each step.

Globals_EIS

FrameType = 1001

The core of the JavaScript routines is found in the Globals_EIS section, that contains buttons with JavaScript functions. These buttons are clicked programmatically by the dashboard activation logic as it starts up. With the exception of `cbtStartUp`, which performs the initialization at the start, the other buttons simply hold function definitions that are imported to the Interactive Reporting Studio object model and made available to be used later. Approximately 100 functions are contained within a dashboard created with Dashboard Studio.

Globals_EIS acts as the backbone of a Dashboard Studio template. Each frame contains a text label called *txlMe* whose text property points to Globals_EIS. Buttons, drop-down lists, and check boxes use Globals_EIS to detect where their generic control handler is imported, and therefore can construct the call in an open and straightforward manner.

All frame controls contain the line of JavaScript:

```
ActiveDocument.Sections[txlMe.Text].Qiq_onControlClick(ActiveSection,this)
```

(In some template versions, `ActiveSection` replaces `this.Parent`. This does not affect the execution of the code).

The generic event broker uses these parameters to route the call to the control handler and determine the action to take.

Note:

It is strongly recommended that no changes are made to Globals_EIS other than turning diagnostic tracing on or off using `cnAD.Qiq_trace is Off` or `cnAD.Qiq_trace is On`.

Custom_Settings

FrameType = 1002

The Custom_Settings section contains a series of text labels that record the settings made in "Step 7: Configure Properties" on page 75. The properties contained in these text labels apply to the Interactive Reporting document as a whole. The section includes properties such as the Home frame, and the status of toolbars, menu bars, and so on.

No JavaScript is contained in this section. The settings and JavaScript are kept apart so they can be modified separately. Oracle or a third party with JavaScript and Interactive Reporting Studio object model skills generally provide the JavaScript, and this is common to all dashboards built from that template. The dashboard developer provides the settings for the dashboard that are specific to the dashboard.

The separation of code and data enables Oracle or third party developers to provide updates to the JavaScript without compromising the settings you make.

Updates to templates and dashboards are provided as an Interactive Reporting document with one or more sections. Old sections are removed by the Dashboard Development Services Update Utility (see "Using the Dashboard Development Services Update Utility" on page 125) and replaced by sections contained in the updated Interactive Reporting document. The Dashboard

Development Services Update Utility is based on the technology that underpins the Import feature or Dashboard Studio Merge Utility. See "Using the Import Feature" on page 117.

1100 Custom

FrameType = 1008

The 1100 Custom section enables you to override the default behavior of most standard dashboard controls.

1100 Custom is hidden each time a dashboard is opened, so use View, then Unhide Section to make it visible before modifying the code.

For each customizable standard control; for example, navigation controls, filter controls, toolbar buttons, and so on, there is a corresponding control in 1100 Custom. The event handlers for these controls define the functions that are used to override the default behavior.

Note:

In the default template, override functions are usually commented-out. When adding code, ensure that the function is called by removing surrounding comment markers.

Two override functions are available:

- Pre-process Function—Called before the standard processing
- Post-process Function—Called after the standard processing

If the pre-process function returns *true,* the standard processing continues and the post-process function is called when the standard processing is complete. If the pre-process function returns *false,* the standard processing is not performed and the post-process function is not called.

It is not necessary to enable pre- and post-process functions for a control as the template checks if a pre- or post- process function is defined before calling it, and a missing pre-process function is assumed to have returned *true,* meaning the standard processing occurs and the post-process function (if defined) is called.

The list illustrates the flow of control:

1. Click a control (for example, `picPrint`).
2. The generic control handler is called.
3. The control handler identifies and calls the specific handler.
4. The specific handler calls the pre-process function for the control in 1100 Custom.
5. If the pre-process function returns *true,* the specific handler continues and calls the post-process function in 1100 Custom before it exits.
6. If the pre-process function returns *false,* the specific handler exits with no further processing.

Custom JavaScript can be placed in the pre- or post-process function depending on the required effect.

The `gtxlLanguageStrings` label is used to contain text that is sensitive to the set dashboard language. The `gtxlLanguageStrings` are used as part of general display or in drop-down lists. The dashboard loads them in the hashed array called *ActiveDocument.Qiq_harConstants.* The format of the label is:

`<text_id1>=<lang1_id>~<lang1_value1>|<lang2_id>~<lang2_value1>`, and so on.

`<text_id2>=<lang1_id>~<lang1_value2>|<lang2_id>~<lang2_value2>`, and so on.

For example,

```
MONTH_JAN=ENUS~Jan|FRFR~Jan|DEDE~Jan|ESES~ene
MONTH_FEB=ENUS~Feb|FRFR~F%E9v|DEDE~Feb|ESES~feb
```

The language values should be escaped so they do not cause problems if they contain embedded separators or other special characters. For example, (=), (~), (|), and carriage returns.

You can use the JavaScript `escape()` function to create strings. For example, if the text is `Ref=danger~trouble|strife`, enter it in the Interactive Reporting Studio Console window:

```
Console.Writeln(escape("Ref=danger~trouble|strife"))
```

You can copy the escaped values from the Console window, which in this case is:

```
Ref%3Ddanger%7Etrouble%7Cstrife
```

The `gtxlToolTip` label is used to hold the text that is to be shown as a floating comment when the specialized control is pressed. The format of the label is identical for `gtxlLanguageStrings`. The dashboard loads them in the hashed array called *ActiveDocument.Qiq_harToolTip.*

Tip:

If you are changing the nature of the operation, consider changing the graphic that is associated with the operation. A detailed discussion of the programming required is outside the scope of this guide.

GUI Frames or Skins

FrameTypes in the range 1100–1199 are reserved for Oracle frames.

FrameTypes in the range 1200–1299 are reserved for third party custom frames.

GUI frames that end users see are derived from frame prototypes. Prototypes exist only in ESMs and are available in the Frame prototype to use drop-down list in “Step 2: Add and Rename Frames” on page 53. These prototypes are selected as the basis for dashboard frames to be created. In runtimes, prototypes are discarded with the Wizard section.

A variety of frame prototypes are available in the standard release, and can be customized in several ways:

- Apply a customized color schemes to a frame layout through “Step 8: Configure Style” on page 79
- Elements such as the navigation or top panel can be moved to achieve alternate layout effects

- Additional graphical elements can be introduced that leverage the properties of current style elements

The third customization type enables you to insert a graphic that uses the style of another frame element; for example, the navigation or top panel styles. This type of customization is achieved by creating graphical objects with the name *customqiq_<element name>*.

For example,

`customqiq_FrameWorkNav`—Leverages the properties of the navigation panel

`customqiq_topPanel`—Leverages the properties of top panel

`customqiq_Background`—Leverages the properties of the display background

In Interactive Reporting Studio, enter Design mode (Ctrl+D), to find the name of a dashboard element, select the element. The name is displayed in the status bar.

Tip:

You can specify an object name as long as it begins with the correct prefix (for example, `customqiq_topPanel`, `customqiq_topPanel2`). The prefix enables you to create multiple elements that inherit an identical set of styles.

User and system controls are available on each GUI frame. The user controls can be set as visible or invisible. The system controls are invisible even in Design mode.

All system controls are locked and sent to back. These settings ensure consistency and prevent the controls from being deleted or moved.

System Controls

System controls are outlined in this topic.

Caution!

If these controls are deleted, the dashboard may stop working correctly.

Allow2D3DSwitching

Holds configuration information for *pic2D* and *pic3D*. Values can be TRUE or FALSE.

AllowChartPivotSwitching

Holds configuration information for *picChart* and *picPivot* . Values can be TRUE or FALSE.

CloneImplements

Blank for frames that are not prototypes. Prototypes tell the Wizard the services that the frames are capable of handling when they are created or copied from the prototype.

drp42D

Not used by standard functions. Custom functions overriding *pic2D* can use this control to hold configuration information.

drp43D

Not used by standard functions. Custom functions overriding *pic3D* can use this control to hold configuration information.

drp4Autosize

Holds configuration information for *picAutosize.* Values can be TRUE or FALSE.

drp4AutoZAxis

Holds configuration information for *picAutoZAxis.* Values can be TRUE or FALSE.

drp4Chart

Not used by standard functions. Custom functions overriding *picChart* can use this control to hold configuration information.

drp4ChartType

Holds configuration information for *picChartType.* Values are the multiple available chart types and can be TRUE or FALSE.

drp4Export

Holds configuration information for *picExport.* Values are the section names that can be exported. If only one value exists, clicking *picExport* calls the export function, otherwise a list of sections is displayed.

drp4Help

Not used by standard functions. Custom functions overriding *picHelp* can use this control to hold configuration information.

drp4Language

Not used by standard functions. Custom functions overriding *picLanguage* can use this control to hold configuration information. The *picLanguage* code acquires values from Globals_EIS.

drp4Legend

Holds configuration information for *picLegend.* Values are the legend controls available and can be TRUE or FALSE.

drp4Pivot

Not used by standard functions. Custom functions overriding *picPivot* can use this control to hold configuration information.

drp4Print

Holds configuration information for *picPrint*. Values are the section names that can be printed. If only one value exists, clicking *picPrint* calls the print function, otherwise a list of sections is displayed.

drp4Word

Not used by standard functions. Custom functions overriding *picWord* can use this control to hold configuration information.

EmbeddedTargets

Holds a map that identifies the section represented by embedded objects; for example;

```
Object~Pivot1|Type~5|Target~Lead Pivot
Object~Table1|Type~4|Target~Top 5 Table
Object~Chart1|Type~6|Target~Revenue Chart
```

It is not possible to discover the embedded object name if using the Interactive Reporting Studio object model. The label is used to construct a set of arrays when the frame is first initialized. Open the Script Editor and use the object browser, the names are illustrated as part of the frame object model.

Programmers that must manipulate underlying charts, pivots, or tables associated with the objects can use these arrays: `Qiq_arrPivot`, `Qiq_arrChart`, and `Qiq_arrTable` these are single dimensional arrays that contain the section names represented on the frame. `Qiq_arrFrameObjects` is a two-dimensional array. These arrays are created when the frame is first activated.

This is an array structure example:

```
Qiq_arrFrameObjects[0][0] = Pivot1
Qiq_arrFrameObjects[0][1] = 5
Qiq_arrFrameObjects[0][2] = Lead Pivot
Qiq_arrFrameObjects[1][0] = Table1
Qiq_arrFrameObjects[1][1] = 4
Qiq_arrFrameObjects[1][2] = Top 5 Table
Qiq_arrFrameObjects[2][0] = Chart1
Qiq_arrFrameObjects[2][1] = 6
Qiq_arrFrameObjects[2][2] = Revenue Chart
```

FrameType

A value in the range 1100–1199 for Oracle frames or a value in the range 1200–1299 for third party frames.

Implements

Keywords that tell the Wizard whether the frame occurs in a step of Dashboard Studio. These keywords include:

ASSOCIATE1

NAVIGATE1

LIMITS1

STYLES1

STARTUP1

FRAMES1

HOME1

PrototypeName

Frame prototype name from which the frame was derived.

txlAction

Control that is currently using *drpAction*. Enables the generic control handler to identify which code to call in response to a click event in *drpAction*.

txlClass

Plug-and-play component that handles the frame. See the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*.

txlCustom

Holds JavaScript that overrides or extends a base control; for example, 1100 Custom.

txlMe

Holds JavaScript that handles base controls; for example, Globals_EIS.

VersionInfo

Contains information used by the version control and update software used by Oracle.

Active Controls

Frames that contain active controls and informational controls are described in "System Controls" on page 209. These are the active controls that cause an action to occur when they are pressed, and they contain an identical line of JavaScript in their event handlers.

```
ActiveDocument.Sections[txlMe.Text].Qiq_onControlClick(this.Parent,this)
```

(In some template versions, `ActiveSection` replaces `this.Parent`. This does not affect the execution of the code).

`QIQ_onControlClick` examines the name of the control (`this.Name`) and calls the handler to deal with it. The process of finding the handler is listed:

1. Standard controls that contain names and handlers are called based on the control name that was clicked. If the name is recognized, the process ends and the remaining steps are ignored.

2. If the name is not recognized, the control is split on the (^) and passed to the Quick Filters handler. It handles controls with names in the form <*filter_name*>^qiq<suffix>. If the control is recognized, it is processed and the remaining steps are ignored.
3. If the handler does not recognize the control, an attempt is made to locate a plug-and-play component (a section of FrameType 1020), by splitting the control name on the (^). For example, *some_name^class_name^suffix* which returns a component name "*class_name*".
4. If the name of the control cannot be split, or a FrameType 1020 cannot be found, the framework looks at `txlClass.Text`. It uses `txlClass.Text` to locate a section of FrameType 1020, and makes the call:

   ```
   ActiveDocument.Sections[txlClass.Text].Qiq_onClick()
   ```
5. If that fails, processing is completed without useful results.

This method provides a low overhead mechanism for incorporating external functionality into the standard frames. The advantage of being part of the framework includes facilities for floating comment and language handling.

Component Frames

FrameType = 1020

These frames play an identical role to frames with FrameType 1008, as they hold custom JavaScript for a component. The distinguishing features between the two FrameTypes are outlined in Table 12.

Table 12 Component Frames

FrameType = 1020	**FrameType = 1008**
Holds JavaScript for all controls for one component (Components can be called by one or more controls)	Holds JavaScript for multiple controls or components (Each control is serviced by the set of six functions)
The framework calls these functions if they are available: ● `Qiq_isSystemSection` ● `Qiq_onWizardApply` ● `Qiq_initToolTip` ● `Qiq_initLanguage` ● `Qiq_onActivate` ● `Qiq_onClick` ● `Qiq_onDeactivate` ● `Qiq_onPostProcess` ● `Qiq_onPreProcess` ● `Qiq_onSelection` ● `Qiq_onShutDown` ● `Qiq_onStartUp` ● `Qiq_reset`	The framework calls one of these six functions: ● `Usr_ctrl_OnActivate` ● `Usr_ctrl_PostActivate` ● `Usr_ctrl_OnClick` ● `Usr_ctrl_PostClick` ● `Usr_drpctrl_OnClick` ● `Usr_drpctrl_PostClick` where `ctrl` and `drpctrl` are the names of a standard control

FrameType = 1020	FrameType = 1008
● Qiq_makeRunTime	

Dashboard Studio components are dashboards with FrameType 1020. One or more sections may be required for the component to function correctly. These must be packaged in the Interactive Reporting document with the component if it is to be distributed.

Components have the advantage of being able to be inserted into other templates with the Dashboard Studio Merge Utility. The Dashboard Studio template framework recognizes these components and automatically includes their services. See "Using the Merge Tab" on page 122.

When the document starts up, the template framework looks for sections with FrameType 1020, and it causes the `OnClick()` event of every `CommandButton` in that section. This event causes all functions in these `CommandButtons` to become instantiated as properties of the dashboard, provided they contain JavaScript of the form:

```
var mySection=this.Parent
function myFunction(myParams){
   // the code of the function
}
mySection.myFunction=myFunction
```

When a control on a frame is clicked, the handler is invoked.

The discovery and recognition of the control is based on the control name or the contents of `txlClass` or the section name with FrameType 1020. The format of these controls is `<controlId>^<SectionName>`.

For example,

```
Section = XyzAmortize, FrameType=1020
Control (on a user frame) = picDoAmort^XyzAmortize
```

When a control is clicked, the generic control handler looks for a list of known controls. If the control is not found, the control name is split on the (^), and a section of FrameType 1020 with an identical suffix is used.

If the control cannot be split or if the 1020 section cannot be located, the generic handler looks for a section based on the content of `txlClass`.

For example,

```
Section = XyzAmortize, FrameType=1020
Control (on a user frame) = picDoAmortize
txlClass = XyzAmortize
```

The two techniques enable you to provide access to a component:

- From general frames that may contain controls that call other classes
- From frames that are primarily serviced by one component; for example, a configuration frame

In general, to enhance the functionality or refine the business rules of a core framework button, code must be added to the button to be refined in a frame of type 1008. However, to create substantial, freestanding functionality that must be used across multiple templates, a component (FrameType = 1020) should be created. Also see the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide.*

14 Diagnostics

In This Chapter

Stack Trace Function

All significant JavaScript routines in Dashboard Studio dashboards call the `Qiq_enter` function on entry and the `Qiq_exit` function on exit. A stack trace of function calls enables you to monitor the development process and identify where and why problems occur.

Execution Stack

Calls to `Qiq_enter` or `Qiq_exit` are made by the standard functions in Dashboard Studio and the execution stack is maintained when the functions start and end. Custom functions that users write are easier to maintain if the same technique is adopted.

The low-level trace adds a small and unnoticable overhead to each function call. The constant low-level tracing enables valuable information to be accessible should a problem arise.

The execution stack is implemented as an array. Whenever a function is called, an entry is added to the array to indicate that the function is active. Whenever code execution returns from a function, the last entry in the array is discarded. If the execution terminates abnormally, the stack shows the function where the error occurred and what other functions were called in the process.

Stack Trace Display

The execution stack is maintained at all times. The stack is empty if the execution terminated as expected, and holds the list of callers if the execution terminates abnormally. A diagnostic report can be exported to Notepad, that includes the stack trace, by executing this JavaScript:

```
ActiveDocument.Sections["QIQ Diagnostics"].Shapes["cbtTrace2Txt"].OnClick()
```

The JavaScript can be executed by entering the syntax into the Interactive Reporting Studio Console window, or by entering it as a permanent Custom menu command.

To make the command permanent in the Custom menu, select Tools, then Customize and add it to the Custom menu provided by Interactive Reporting Studio.

Note:

It is recommended that the Console window is open while the dashboard is running, to report errors that occur.

Extract 1

This extract displays a stack trace report with no errors.

```
======== Dashboard Studio - start of trace log ========
        Sunday, August 07, 2002 23:48:39

======== Top of execution stack =======================================

======== Bottom of execution stack ====================================
       Full Classes function audit trail requested – see the end for trace
level details

======= Start of function audit trail =============================
====== End of function audit trail ================================

        Trace level details


level = [5] for Class 1100 Custom
level = [5] for Class Qiq_ChartView
level = [5] for Class Qiq_RelatedDocuments
level = [5] for Class Qiq_SwitchChartPivot
level = [5] for Class Qiq_cal
level = [5] for Class Qiq_chartspotlight
level = [5] for Class Qiq_iServerExport
level = [5] for Class Qiq_limits
level = [5] for Class Qiq_picker
level = [5] for Class hysl_expList
level = [5] for Class hysl_hideFacts
level = [5] for Class qiqsort
level = [5] for Class qiqsl
level = [5] for Class Globals_EIS
level = [5] for Class Others
======== Dashboard Studio - end of trace log ========
```

Extract 2

This extract displays a stack trace report with an error.

```
======== Dashboard Studio - start of trace log ========
        Sunday, August 07, 2002 23:48:39

======== Start stack trace =======================================

[03]In Qiq_documentLimits ()
[02]In Qiq_applyCbxLimit()
```

```
[01]In Qiq_quickOnControlClick(Team KPIs by year, Team^qiqcbx2)
[00]In Qiq_OnControlClick(Team KPIs by year, Team^qiqcbx2)

======== End stack trace ========================================

        Full Classes function audit trail requested

======= Start of function audit trail =============================
====== End of function audit trail ================================

        Trace level details

level = [5] for Class Qiq_ChartSpotlight
level = [5] for Class Qiq_metaData
level = [5] for Class Qiq_queryLimits
level = [5] for Class Qiq_sortPivot
level = [5] for Class Globals_EIS
level = [5] for Class Wizard
level = [5] for Class Others
======== Dashboard Studio - end of trace log ========
```

In Extract 2, entries are present between the start and end of the stack trace log.

The chain of events starts with the calling of the `Qiq_OnControlClick` function > `Qiq_quickOnControlClick` function > `Qiq_applyCbxLimit` > `Qiq_documentLimits`. As this function is at the top of stack trace, it is a good indication that the problem occurred in that function.

Note:

The use of “>” in the example, represents the flow of events and the calling of the next function.

Implement Tracing

Stack tracing is only as good as the discipline used by the programmers of the dashboard. Stack trace relies on each function calling `Qiq_trace` at the beginning and just before the end. The sample code illustrates how code should be written:

```
function myFunction(){
  Qiq_enter(“myFunction”,1,this.Parent)
  for (var a=1;a<=this.Parent.Shapes.Count;a++){
    Qiq_trace(“looking at shape “+a,2,this.Parent)
    //
  }
  //
  Qiq_exit(“myFunction”,1,this.Parent)
}
```

`Qiq_trace` takes these three parameters:

A String

Added to the stack. If the string starts with *In* followed by a space, it is assumed to be the start of the function. If the string starts with *Out* followed by a space, it is assumed to be the end of the function. If it does not start with *In* or *Out* it is assumed to be an inline call, and is not recorded in the stack trace. Inline calls are written to the trace report if tracing is turned on for the component and the second parameter is beneath the trace level requested.

A Number

Indicates the trace level for the call to be displayed in the full trace report or log. A stack trace is maintained no matter what trace level is selected. For example, the entry and exit from the function is reported if the trace level is 1 or greater. The iterations of the *for loop* are reported only if the trace level is 2 or greater. See "Full Tracing" on page 220.

A Section Identifier in which the Code is Located

JavaScript code is held inside controls such as buttons, check boxes, and so on. `this` refers to the control in whose event handlers the code is entered and `.Parent` refers to the parent object in which the object exists. The shorthand is recommended over the full and explicit naming of the section, because it enables you to rename the section or object, or move JavaScript to another section without having to change the code to reflect the changes made.

Note:

A trace statement must be added to all points in a function where the code execution may exit. The JavaScript *return* statement causes an early exit. All statements following *return* are not executed. Thus, the call to the `Qiq_trace` should be placed on the line immediately before a *return* statement.

Full Tracing

Turn tracing on if you want to examine the full flow of execution. Tracing imposes a significant overhead on the system unlike the stack trace, which shows only the active functions. Depending on the trace level required, Dashboard Studio can keep track of all calls made which are then available for output to Notepad by using the command; for example

```
ActiveDocument.Sections["QIQ Diagnostics"].Shapes["cbtTrace2Txt"].OnClick()
```

The history is kept in a list which increases in size until tracing is turned off or the dashboard is closed. If a dashboard is saved and closed with tracing on, tracing resumes when it is reopened.

These commands can be made part of the Custom menu environment to turn tracing on and show the tracing window:

```
var x=ActiveDocument.Sections["QIQ Diagnostics"];
x.cbxTraceOn.Checked=true;
x.cbxTraceOn.OnClick();
if(!x.Visible){x.Visible=true;x.Activate()}
```

These commands can be incorporated into the Custom menu environment to turn tracing off:

```
var x=ActiveDocument.Sections["QIQ Diagnostics"];
x.cbxTraceOn.Checked=false;
x.cbxTraceOn.OnClick()
```

Using the QIQ Diagnostics Frame

QIQ Diagnostics provides a convenient way of identifying the level of tracing to perform (that is, how much trace history to display). The commands discussed earlier are programmatic clicks on the various controls in QIQ Diagnostics.

Access QIQ Diagnostics, by clicking on the Copyright frame, or by selecting View, then Unhide Section, and selecting QIQ Diagnostics.

➤ To use QIQ Diagnostics:

1. **In QIQ Diagnostics, select Start Trace.**
2. **Click Refresh List to show a full list of class sections in Components in BQY.**

   QIQ Diagnostics performs levels of tracing for each section. The recorded tracing information increases as the number increases.
3. **Select an option to set the trace level.**
4. **Move selected sections between Components in BQY and Components to Trace by clicking > and <.**
5. **Navigate to another frame and return to refresh QIQ Diagnostics.**

   The trace information is displayed.

   Various buttons and controls are available on the frame. These can be divided into two levels: General and Advanced Diagnostics.

General Diagnostics

These are the buttons and controls for general diagnostics.

- Trace to Console—Display trace information in the Console window and QIQ Diagnostics
- Start Timer—Provide timing information and check the internal performance of sections
- Add Comment—Add text to the trace information
- Show Execute Window—Select to display the Execute and edit JavaScript code window (This window replaces Log Find Tools. See "Execute and Edit JavaScript Code" on page 222)
- Show Log in Notepad—Export the trace information as a text file into Notepad (The text file name is identical to the Interactive Reporting document and is located with the Interactive Reporting document)
- Show Stack Trace—Display the current stack trace
- Show FrameTypes—Display section information
- Show Complexity—Summarize the dashboard and compile section statistics; for example, the number of objects by type, number of rows, and so on (The summary information can be saved into Excel and used to evaluate the complexity of the Interactive Reporting

document, and then can be imported into Interactive Reporting Studio for further analysis of the dashboard)

- Show Timings—Add the timing information to the trace log within the Interactive Reporting document as a text file that EPM Workspace can read
- Clear Log—Clear all text in the list or stack trace
- Clear Stack Trace—Clear the stack trace
- Clear Timer—Clear the timer information

Advanced Diagnostics

The advanced diagnostic facilities:

- Log Find Tools
- Execute and Edit JavaScript Code
- Object Model Inspector

Log Find Tools

The log find tools window is displayed if Show Execute Window is not selected.

These tools include buttons and a control.

- Find Name—Locate function names in the log (Enter a name and select Partial Match to match a function that begins with the entered name. Match results are added to the list, and corresponding entries are highlighted in the list next to the General Diagnostics controls)
- Find Break—Locate a discontinuity in the log (Functions begin by logging an *In* call and exit with an *Out* call. If an error occurs, the *Out* call will be missing, and execution will stop. However, execution may continue, so the end of the log does not show the function in error. Find Break analyzes the logs and identifies the *In* calls with missing *Out* calls, and the sequence of calls that led to the *In* call. The functions that led to the discontinuity are displayed in the list, and corresponding log entries are highlighted in the log)
- Find Out—Locate the *Out* call that corresponds with the selected *In* call (Calls are matched if they share a function name, an identical call level (indentation), and if no intervening *In* calls are made to the same function. If the match fails, the rules are relaxed. The call level (indentation) and *Out* calls are matched even if they are not at the same call level. The relaxing of rules occurs if a log discontinuity exists, then the apparent level of the *Out* call is different to the *In* call)

Execute and Edit JavaScript Code

The Execute and edit JavaScript code window is displayed if Show Execute Window is selected.

When developing and debugging scripts on the desktop client, dashboard developers enter JavaScript into the execute window and call it to see how it functions. Scripts created in the execution window can be stored, saved, and recalled. Script execution results are displayed in the trace text box above the Execute and edit JavaScript code window.

After entering script into the window, use these buttons and controls to save, reuse, and test scripts:

- Select a script—Displays the scripts saved by name
- Save Code—Saves the script with a name entered in the text box
- Call Code—Executes the script that is displayed in the execute window (Enter code or select a script from the drop-down list and click Call Code to execute the script. The syntax, results, and errors are displayed in the trace text box)
- Delete Code—Removes saved scripts from the Select a script drop-down list

Object Model Inspector

The Inspector includes a set of controls that display the content and values of the object model at different levels of the object tree.

- Sort by Type—If not selected, Properties and Objects under inspection are sorted by name (If selected, Properties are displayed, then the Objects, and finally the functions. All are sorted by name)
- Application—Display members of the Application object (To view the Application object members, select the line in the Inspector, and click ↘. Only members of objects are available to view)
- ActiveDocument—Display members of the ActiveDocument object
- Show Component—List Dashboard Development Services component sections with FrameType 1020 (Provides a shortcut that bypasses searching ActiveDocument sections for components. When debugging it is useful to view members of components)
- Show Frames—List Oracle's Hyperion® Dashboard Development Services sections that are frames with FrameType 1100–1299, and that are sections visible to the user (When debugging it is useful to view members of frames)
- ↖ (Up)—Navigate to the parent of the object (The parent object is displayed in the Inspector and the path is updated to reflect movement up the hierarchy)
- ↘ (Down)—Navigate to view members of the listed object (Select a line in the Inspector with an object, and click ↘. The selected object is displayed in the Inspector and the path is updated to reflect movement down the hierarchy)

Diagnostic Sample Output

This sample output is the full diagnostic trace for the operation that returned an error earlier in Extract 2. In the earlier output, tracing was not operational, so only the stack trace was logged. In this example, tracing is on and the operation that caused the error is repeated.

```
======== Dashboard Studio - start of trace log ========
        Tuesday, August 20, 2002 17:07:38
======== Start stack trace =======================================

[03]In Qiq_documentLimits()
[02]In Qiq_applyCbxLimit()
[01]In Qiq_quickOnControlClick(Revenue by Type,Year^qiqcbx7)
[00]In Qiq_OnControlClick(Revenue by Type,Year^qiqcbx7)

======== End stack trace =======================================

 Full Classes function audit trail requested - see the end for trace level
details

======== Start of function audit trail ===============================
17:07:31[1]=> In Qiq_OnControlClick(Revenue by Type,Year^qiqcbx7)
17:07:31[1]=>   In Qiq_quickOnControlClick(Revenue by Type)
17:07:31[1]=>      In Qiq_findLimits(Year)
17:07:31[1]=>      Out Qiq_findLimits()
17:07:31[1]=>      In Qiq_applyCbxLimit()
17:07:31[1]=>      In Qiq_findLimits(Year)
17:07:31[1]=>      Out Qiq_findLimits()
17:07:32[1]=>      In Qiq_ClearSelections(lbxSelected)
17:07:32[1]=>      Out QIQ_ClearSelections
17:07:32[1]=>      In Qiq_documentLimits()
======== End of function audit trail ===============================

        Trace level details

level = [5] for class Qiq_chartSpotlight
level = [5] for class Qiq_metaData
level = [5] for class Qiq_queryLimits
level = [5] for class Qiq_sortPivot
level = [5] for class Globals_EIS
level = [5] for class Wizard
level = [5] for class Others

======== Dashboard Studio - end of trace log =======
```

The time of each call to `Qiq_trace` is recorded as part of the trace in the form hh:mm:ss.

The first parameter to `Qiq_trace` is indented to illustrate the depth of the stack trace. Each *In* call `(Qiq_enter)` to `Qiq_trace` indents and each *Out* call `(Qiq_exit)` to `Qiq_trace` reverses the indentation.

The second parameter to `Qiq_trace` is shown in []. In the example, all trace calls are at trace level 1. Therefore, if you request trace level 1 or greater in the dashboard class section that contains JavaScript, the output will contain these trace calls.

Using the Type Library File to Synchronize Releases

The Type Library file (TLB) describes the interface offered by a COM server such as Interactive Reporting Studio. Dashboard Studio uses the TLB to communicate with the services that Interactive Reporting Studio offers. In a standard installation, the TLB is located in the Windows system32 folder and is called *brioqry.tlb*. The Interactive Reporting Studio installer keeps the application and the TLB in synch. However, when non–standard mechanisms for upgrading are used problems may arise.

➤ To view **Installation Diagnostics**, click on the **Select a Framework Template** window of Dashboard Studio wizard.

15 Configuring and Embedding Reports

In This Chapter

About Reports

Depending upon the Interactive Reporting Studio release that you are using, you can configure reports to be printed or exported, or you can embed reports in a dashboard frame.

Reports can contain BLOB images from a database or aggregate tables. For information on BLOB image support, see the *Hyperion Interactive Reporting Studio User's Guide.*

Configuring Reports

In Interactive Reporting Studio, you can add reports to the list of objects to print or export from a frame. These reports may not be a printer-friendly version of a dashboard; however, you may want to print reports to exclude dashboard elements such as the button bar. By design, only objects that are dragged onto a frame can be printed or exported from frames. As reports in these earlier releases cannot be dragged onto the frames, these sections are not typically available from the print or export drop-down lists.

➤ To manually add reports to the list of printable or exportable objects:

1 **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2 **Create a *dummy* section.**

For example, a chart, pivot, or table (with no columns added).

3 **Drag the *dummy* section onto the dashboard frame from which to print or export reports.**

4 **Double-click the section.**

Properties is displayed.

5 **Enter a name.**

For example, enter `Report1`.

6. **Repeat steps 1–5, to configure a *dummy* section for each report to print or export, and give each section a unique name.**

7. **Select a *dummy* section, shrink it to the smallest size, and send it to the back of the frame.**

8. **In Dashboard Studio, navigate to Step 4: Associate Frames with Charts, Pivots, or Tables.**

   Reports are displayed in Frames and Embedded Objects list.

9. **From Frames and Embedded Objects, select a report, to associate with the *dummy* section.**

   Select One Section, on the right of Step 4: Associate Frames with Charts, Pivots, or Tables, is displayed.

   Ensure Allow Printing of Object and Allow Export of Object are selected.

10. **After all *dummy* reports are associated, click ✓ to apply selections to the dashboard.**

    The reports are available when 🖨 or ➡ is clicked to print or export.

Embedding Reports

Interactive Reporting Studio reports can be embedded into dashboards and directly printed or exported.

Interactive Reporting Studio and Interactive Reporting Web Client enable reports to be embedded at design time. Use the vertical scrollbar to navigate through an embedded report in Interactive Reporting Studio or Oracle's Hyperion® Interactive Reporting Web Client. In Oracle Enterprise Performance Management Workspace, Fusion Edition, navigate by using the paging and navigation toolbar.

➤ To embed reports:

1. **In Interactive Reporting Studio, press Ctrl+D to enter Design mode.**

2. **From the catalog pane, expand Reports.**

3. **Select and drag the report onto the dashboard frame.**

4. **Double-click the report.**

   Properties is displayed.

5. **Enter a name.**

   For example, enter `Report1`.

   If you are embedding a report into a Webdash document using the Build Dashboard frame, the name of the report must begin with Pivot to be recognized as a tabular structure. See "Embedding Sections" on page 103.

6. **In Dashboard Studio, navigate to Step 4: Associate Frames with Charts, Pivots, or Tables.**

   Reports are displayed in Frames and Embedded Objects list.

7. **From Frames and Embedded Objects, select the report.**

Select One Section, on the right of Step 4: Associate Frames with Charts, Pivots, or Tables, is displayed.

Ensure Allow Printing of Object and Allow Export of Object are selected.

8 **Optional:** If the report is not visible in **Frames and Embedded Objects**, click ⟳.

9 Click ✓ to apply selections to the dashboard.

The reports are available when 🖨 or ➡ is clicked to print or export.

16 Customizing and Configuring Templates

In This Chapter

About Customization

Dashboard Studio provides several templates. The basic template (`Base_Template.bqy`), by default, contains frames that incorporate a simple color scheme and no embedded graphics. Because Dashboard Studio is built on a fully extensible framework, templates can be customized to suit the needs of an organization.

Three levels of template customization are available to use.

- Cosmetic customizations are made by setting color schemes and adding logos to frames
- Customize a feature by extending or changing the functionality
- Create a feature, called a component, using advanced customization techniques (see the *Hyperion Interactive Reporting – Object Model and Dashboard Development Services Developer's Guide, Volume 7: Component Reference Guide*)

Note:

To customize the QIQ Base frame prototype use the Home frame.

Multiple-Release Warning

If your organization uses only one release of Interactive Reporting Studio, you can skip this warning. Interactive Reporting Studio is backwards compatible; thus a later release opens Interactive Reporting documents built with an earlier release. However, if an earlier release opens an Interactive Reporting document created with a later release, the earlier release discards information about features that it does not recognize. If the Interactive Reporting document is saved, the unrecognized information is lost permanently. Therefore, in a mixed-release environment, use only features that are supported by the earliest release. For example, if your organization uses Release 6 and Release 8, do not use UserValues (introduced in Release 8). If your organization uses Releases earlier than 8.2 and Release 8.2 or later, do not use JPEG or GIF images, which are not recognized and are discarded by Releases 8.1 or earlier. Also see "Compatibility Between Releases" on page 16.

Adding Logos and Color Schemes

This topic looks at the first level of customization. It is recommended that templates are customized before they are released to end users, to save time and ensure consistency.

Customizing a template requires these steps:

1. Open a template.
2. Make customizations.
3. Save customized template to the Templates folder.
4. Include customized template in Databoard Studio.

➤ To customize a template:

1 In Interactive Reporting Studio, open a template to customize.

2 Open Dashboard Studio.

Select a Framework Template is displayed.

3 Select **Connect to open dashboard**, select the template, and click > to move to the next step.

A dialog box is displayed warning that a template is to be edited directly.

4 Click **OK**.

5 Navigate to **Step 8: Configure Style** of Dashboard Studio, to set the color and style elements.

Minimize Dashboard Studio to work on the template.

6 In Interactive Reporting Studio, press **Ctrl+D** to enter Design mode.

7 From the catalog pane, expand **Graphics**, and drag graphical elements onto the template.

For example, add a company logo by dragging the picture icon onto the left corner of the top panel.

8 In **Select Image**, locate of the logo, and click **Open**.

9 Repeat steps 7–8 to add other images and configure styles.

10 In Dashboard Studio, click 💾 to save the style.

Save As is displayed.

11 Save the customized template back to the **Templates** folder with a unique name.

Three main reasons to save a template with a unique name:

- If Dashboard Studio is uninstalled, templates are deleted (A unique name prevents customized templates from being removed)
- A unique name prevents saving over the original template

- A unique and meaningful name is easily recognizable to business users

A dialog box is displayed to confirm that the Interactive Reporting document is to be saved as a template.

12 Click **Yes.**

The customized template is saved.

Adding Custom Templates to Dashboard Studio

The final step in setting up templates, is to specify the templates that end users can connect to through Dashboard Studio.

In the example, use the template that was customized in the Adding Logos and Color Schemes procedure. Ensure the template is saved and closed.

➤ To add a custom template to Dashboard Studio:

1 **Open Dashboard Studio.**

2 From **Select a Template Framework**, click ☑.

Options is displayed.

3 Click **Include.**

Select the customized template to be available in Dashboard Studio.

4 **Optional:** To provide an alias for the template, perform an action:

- Double-click the template name in Templates, and edit the name
- Select the template name in Templates, and click F2 to edit the name

5 Click **OK.**

The customized template is available to create dashboards.

17 Creating and Configuring Frame Prototypes

In This Chapter

Creating Frame Prototypes

Dashboard Studio ships with assorted frame prototypes. A frame prototype determines the layout and features available for each dashboard frame. Templates can contain several frame prototypes.

This topic looks at how to create a frame prototype. Dashboard Studio templates contain frame prototypes that act as *factories* from which frames are duplicated. Dashboard frames are created in "Step 2: Add and Rename Frames" on page 53.

Frame layouts can be customized by changing frames or creating frame types. The most common method involves changing a frame prototype.

The key elements in a frame prototype are these text labels:

- `FrameType` (values 1100–1299)
- `CloneImplements`
- `PrototypeName`

See "Template Structure" on page 205, for more information the key elements.

Note:

The `FrameType` for prototypes must be unique in an Oracle's Hyperion® Interactive Reporting document. However, the `FrameType` for frames may not be unique.

Table 13 provides a comparison between frame prototypes and frames.

Table 13 Comparison Between Frame Prototypes and Frames

For a Frame Prototype	For a Frame
`CloneImplements != “”`	`CloneImplements == “”`

For a Frame Prototype	For a Frame
`PrototypeName == “”`	`PrototypeName != “”`

The `CloneImplements` property describes the part of the frame that is duplicated or *cloned* and the role it plays. Dashboard Studio detects the property in the dashboard building process. The `PrototypeName` property is the frame prototype name.

Frame prototypes are listed in “Step 2: Add and Rename Frames” on page 53 of Dashboard Studio, in the Frame prototype to use drop-down list.

Duplicating Frame Prototypes

The first step in creating a frame prototype is to create a duplicate. The Square Tab frame prototype in the `Express_Template.bqy`, is used as an example in this procedure.

➤ To duplicate the Square Tab frame prototype:

1 In Interactive Reporting Studio, navigate to and open `Express_Template.bqy` from the installation directory.

2 From **Sections**, right-click the Square Tab frame prototype, and select **Duplicate Section.**

Square Tab2 is created.

3 Right-click Square Tab2, and select **Rename Section.**

4 In **Section Label**, enter `Square Tab Wide`.

5 Click **OK.**

The prototype is duplicated.

Editing Frame Prototypes

Edit the duplicated frame, to suit your requirements.

➤ To edit a frame prototype:

1 Use the dashboard from the Duplicating Frame Prototypes procedure, and press **Ctrl+D** to enter Design mode.

2 Select the side-label navigation panel which contains Quick Filters, right-click, and select **Properties.**

Properties is displayed.

3 Clear **Locked** and click **OK.**

4 Drag the Quick Filters list called *Some_Filter^qiqlbx^showname^a* to the right of the Quick Filters drop-down list.

5 Extend the side-label navigation panel to the right, beyond the Quick Filters list.

6 Double-click the side-label navigation panel.

Properties is displayed.

7. Select **Locked**, and click **OK**.
8. Press **Ctrl+D** to exit Design mode.
9. Save the frame prototype.

   The prototype is customized.

Applying Design Guides in Wide View Mode

In Interactive Reporting Studio, the Square Tab frame prototype in the `Express_Template.bqy`, provides layout design guides to help you use the Dashboard Studio Wide View and Standard View feature. The feature enables a dashboard developer to construct a dashboard with wide objects that are positioned on the left and that hide the side-label navigation panel, or with narrower, standard-sized replicas that are positioned to the right of the side-label navigation panel, and that provide access to the Quick Filters. See "Wide View Properties " on page 91, for information on configuring the Wide View mode feature.

The Square Tab prototype is configured for 1, 2, or 4 visible objects. Table 14 describes the configuration process.

Table 14 Configuration Process

Switch Between Chart and Pivot	Number of Visible Objects	Switch Between Wide and Standard View	Number of Objects	Description
N	1	N	1	1 tall and broad visible object
Y	1	N	2	
N	1	Y	2	
Y	1	Y	4	
N	2	N	2	2 tall and slim visible objects or 2 short and broad visible objects
Y	2	N	4	
N	2	Y	4	
Y	2	Y	8	
N	4	N	4	4 short and slim visible objects
Y	4	N	8	
N	4	Y	8	
Y	4	Y	16	

Modifying the Width of the Side-Label Navigation Panel

To modify the width of the side-label navigation panel, the design guides require change to accommodate the adjusted width.

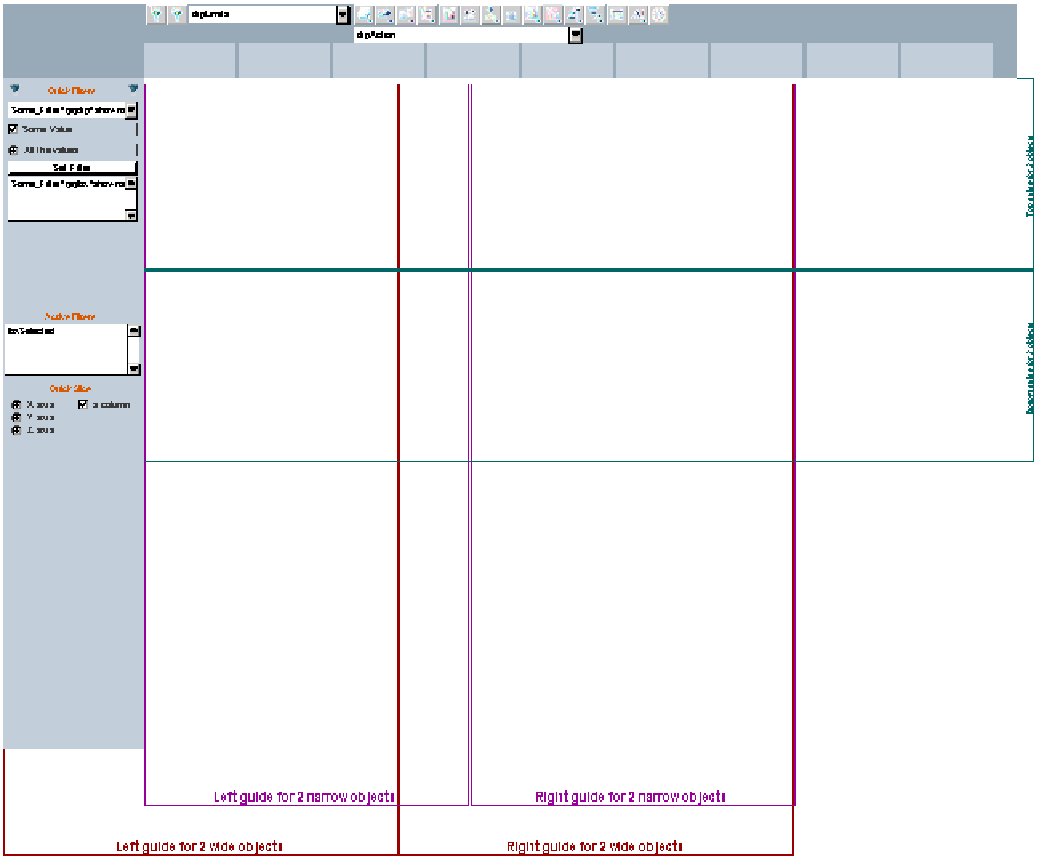
➤ To change the design guides to adapt to the width of the side-label navigation panel:

1 **Use the dashboard from the Editing Frame Prototypes procedure, and press Ctrl+D to enter Design mode.**

2 **Select View, then Zoom to 50%.**

The design guides are carefully layered to be accessible for selection, and are described, from back-to-front, in Table 15.

The shapes at the back protrude from the bottom or to the right so they can be selected. After dragging a chart, pivot, or table onto the dashboard, select a design guide object and align it to the left or right, or make it identical in height and width.


Quick Filters
Some Value
All the values
Set Filter
Active Filters
Quick Slice
X axis
Y axis
Z axis
Left guide for 2 narrow objects
Right guide for 2 narrow objects
Left guide for 2 wide objects
Right guide for 2 wide objects

3 **Double-click Left guide for 2 narrow objects.**

Properties is displayed.

4 **Clear Locked, and click OK.**

5 **Double-click Right guide for 2 narrow objects.**

Properties is displayed.

6 **Clear Locked, and click OK.**

7 Select the unlocked guides, and drag the middle resize handle on left guide, until it is approximately halfway to the right edge of the side-label navigation panel.

8 Press **Ctrl+D** to exit Design mode.

9 Save the dashboard.

The side-label navigation panel width is extended.

Table 15 Design Guide Layers

Shape Name	Line Color	Label Text Value
snapTo_right_2wide	brown	Right guide for 2 wide objects
snapTo_left_2wide	brown	Left guide for 2 wide objects
snapTo_left_2narrow	mauve	Left guide for 2 narrow objects
snapTo_right_2narrow	mauve	Right guide for 2 narrow objects
snapTo_bottom_2	green	Bottom guide for 2 objects
snapTo_top_2	green	Top guide for 2 objects

Moving the Design Guides Accurately

To achieve an accurate result in moving the design guides, repeat these steps until a precise outcome is attained.

➤ To accurately align the design guides:

1 Use the dashboard from the Modifying the Width of the Side-Label Navigation Panel procedure, and press **Ctrl+D** to enter Design mode.

The view remains zoomed to 50%.

2 Unlock the left and right guides, by right-clicking them individually, and selecting **Properties.**

Properties is displayed.

3 Clear **Locked**, and click **OK.**

4 Drag the left guide to the right, and perform these actions:

a. Stop moving when the right of the left guide just overlaps the left of the right guide.

b. When the sizes and alignment are correct, move to step 5.

5 With the guides selected, select **View**, then **Zoom 200%**, for a precise view.

a. Scroll and pan until the left middle resize handle comes into view.

b. Drag the left middle resize handle until it is halfway to the right edge of the side-label navigation panel.

6 Select **View**, then **Zoom 50%.**

7 Select the left guide, right-click, select **Properties.**

Properties is displayed.

8. Select **Locked**, and click **OK.**
9. Select the right guide, right-click, select **Properties.**

   Properties is displayed.

10. Select **Locked**, and click **OK.**
11. Select **View**, then **Zoom 100%.**
12. Press **Ctrl+D** to exit Design mode, and save the dashboard.

    The design guides are precisely aligned.

    The side-label navigation panel can be customized further by duplicating Quick Filters and aligning Active Filters.

➤ To duplicate Quick Filters and align Active Filters:

1. Use the dashboard from the Moving the Design Guides Accurately procedure, and press **Ctrl+D** to enter Design mode.
2. Select the right Quick Filters icon, , and drag it to the right of the side-label navigation panel.
3. Resize the Quick Filters heading.
4. Select and drag the list called *`Some_Filter^qiqlbx^showname^a`*, back to the original location under **Set Filter.**
5. On the side-label navigation panel, use **Ctrl+click** to select these items, the drop-down list, check box, option and command buttons, and the list.
6. Drag the selected objects to the right to create duplicates.
7. Resize `lbxSelected` and the **Active Filters** heading to fit the width.
8. Press **Ctrl+D** to exit Design mode, and save the file.

Verifying the Performance of the Frame Prototype in Dashboard Studio

Verify that the frame prototype performs as required in Dashboard Studio.

➤ To test the frame prototype in Dashboard Studio:

1. Use the dashboard from the Moving the Design Guides Accurately procedure.
2. In Dashboard Studio, navigate to Step 2: Add and Rename Frames.

   The expected behavior is that the Square Tabs Wide frame prototype is available from Frame Prototype to use.

Configuring Multi-Panel Frame Prototypes

The multi-panel frame prototype requires unique treatment as objects must be correctly associated for the panel heading feature to function properly. See "Step 4: Associate Frames with Charts, Pivots, or Tables" on page 68.

➤ To drag, associate, and configure charts with a multi-panel frame prototype:

1. **Start Interactive Reporting Studio, and, on the Welcome dialog, click Cancel.**

2. **Open Dashboard Studio.**

   Select a Framework Template is displayed.

3. **Click → to import a template that contains multi-panel frames into your document.**

   For example, use the Round Tab Panel Template.bqy. See "Using the Import Tab" on page 118.

4. **In Dashboard Studio, navigate to Step 2: Add and Rename Frames.**

   Rename the charts if it is possible that a conflict may occur in a future step.

5. **Select a Frame prototype to use, and click Add.**

   For example, select 4 Panels Round Tab.

6. **Set the name of the frame prototype.**

7. **Move to Step 3: Drag Pivots, Charts, and Tables onto Frames.**

8. **In Interactive Reporting Studio, drag the charts onto the panels in the frame.**

   For example, using the 4 Panels Round Tab frame layout, drag Chart1 onto panel Heading 1, Chart2 onto Heading 2, and Chart3 onto Heading 3.

9. **In Dashboard Studio, navigate to Step 4: Associate Frames with Charts, Pivots, or Tables, and click to associate the charts with the corresponding frames.**

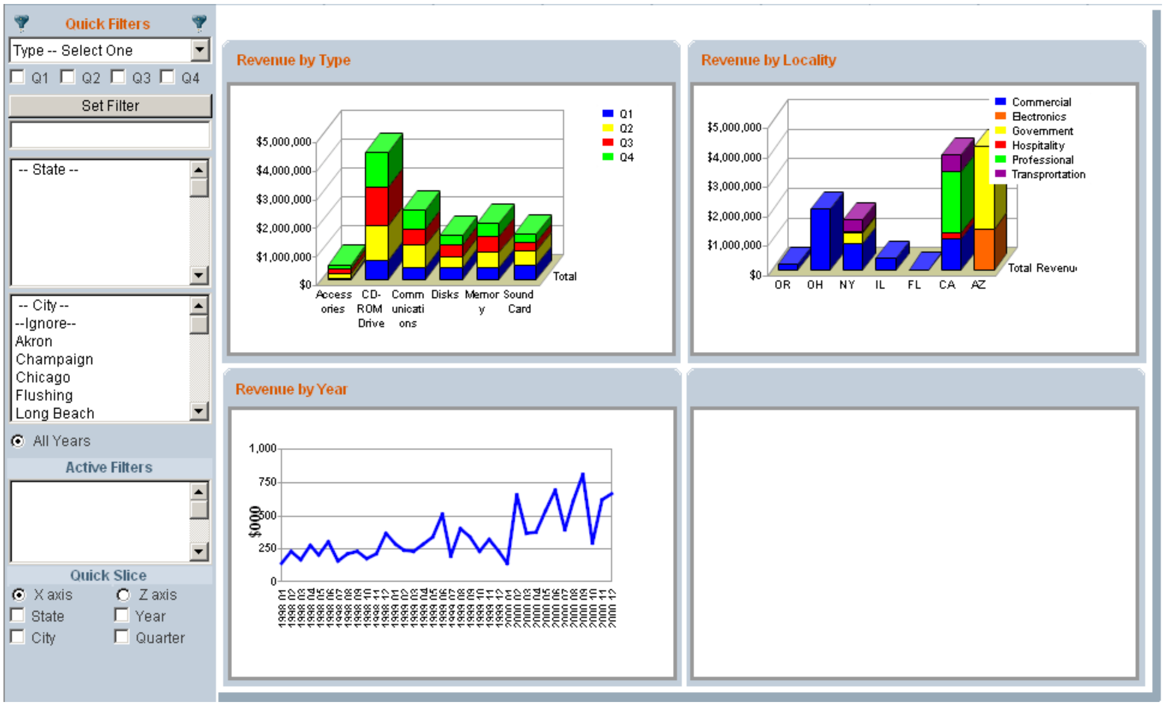
10. **Click ✓ to apply the selections.**

11. **Navigate to Step 7: Configure Properties of Dashboard Studio, and configure the Chart Titles.**

    For example, select *Show all* from the drop-down list.

    Applying the Chart Titles option results in the chart name being displayed in the panel heading. Charts must be correctly associated in Step 4 of Dashboard Studio for the panel heading feature to work. For example, in Figure 9, three charts display corresponding headings. However, Heading 4 is cleared, because no chart is associated with it.

Figure 9 Charts and Panel Headings Configured to Match



12 In Dashboard Studio, click ✓ to apply the configuration.

13 Complete the remaining Dashboard Studio steps to configure and save the multi-panel frame dashboard.

18

Adding and Configuring Calendars

In This Chapter

Adding Calendar Controls

Use these steps to add the calendar control to a dashboard.

➤ To add the calendar control:

1 **In Interactive Reporting Studio, select View, then Unhide Section.**

2 **Select Qiq_cal_UI, click OK, and press Ctrl+D to enter Design mode.**

3 **Drag a rectangle with the mouse around the calendar objects to select all.**

4 **Press Ctrl+C to copy to the objects.**

5 **Navigate to another frame, and select View, then Zoom 50%.**

6 **From the catalog pane, expand Controls, drag a button onto the frame, and position it below the level of the lowest frame object.**

This button is created as a marker object for pasting the calendar objects.

When a dashboard frame is selected in Design mode, or an object is dragged onto a frame, a dotted rectangle displays the furthest boundaries within which objects exist. Ensure the marker is dragged below or to the right of the boundary, so all calendar images will be available for selection.

7 **Select the marker and press Ctrl+V to paste the calendar objects.**

Selecting the marker ensures the pasted objects are positioned below the marker.

8 **Select View, then Zoom 100%.**

9 **Select Clear, and move it to the top right of the calendar.**

10 **Resize the marker to match the height and width of Clear.**

11 **Align the marker to the left of the calendar, level with Clear.**

12 **Double-click the marker, and rename it as `cbtSet_1`.**

13 **Enter `Set` as the Title.**

14 **Click OK.**

Proceed to Duplicating and Positioning Calendars.

Duplicating and Positioning Calendars

Duplicate the calendar objects and position the two calendars on the side-label navigation panel.

➤ To duplicate and position the calendar:

1 **Use the calendar from the Adding Calendar Controls procedure, and select View, then Zoom 50%.**

2 **Drag a rectangle with the mouse again, around the shapes; the calendar and buttons.**

3 **Press Ctrl, and click the selected object. Without releasing the mouse drag the calendar to create a duplicate.**

Ensure the calendar and the duplicate do not overlap.

4 **Select View, then Zoom 100%, and scroll to locate the calendars.**

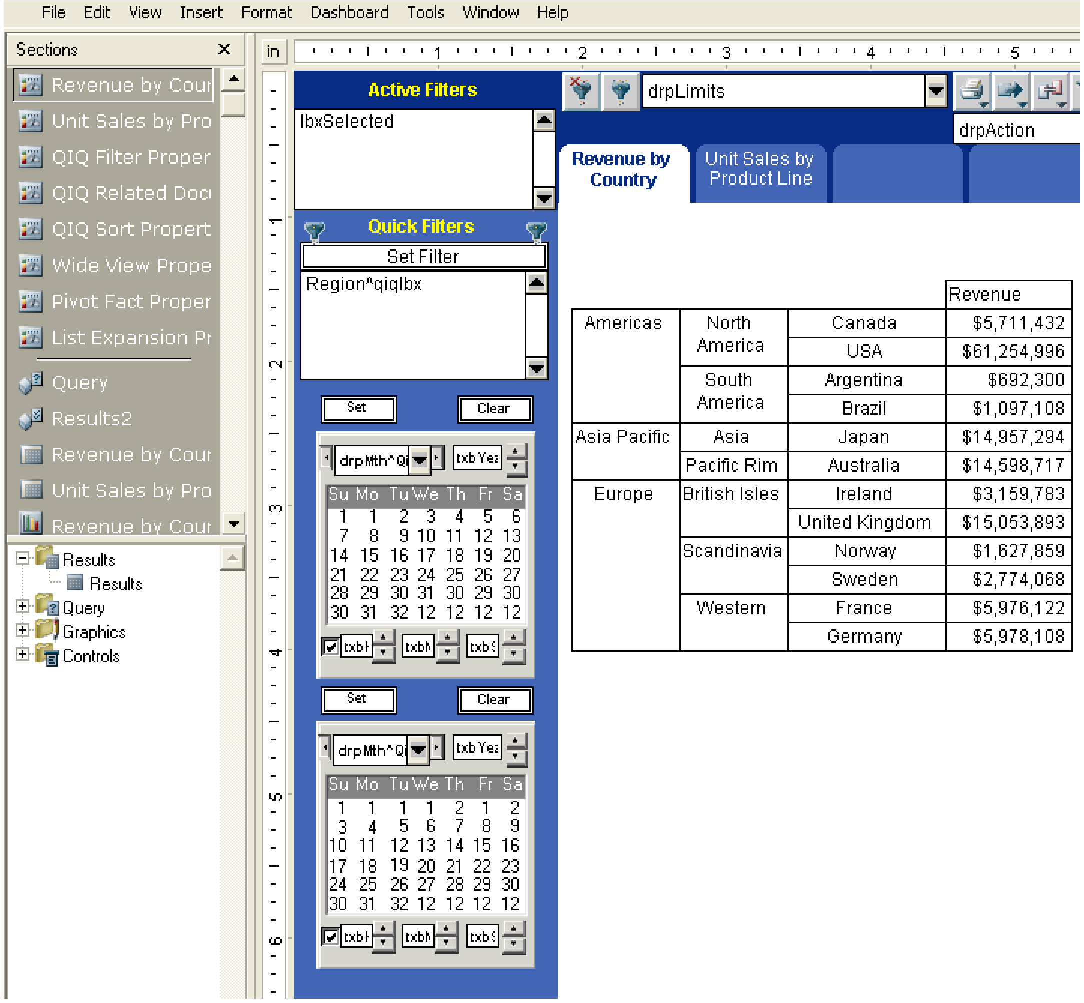
5 **Drag a rectangle with the mouse, around the second calendar, to select all objects.**

6 **Move the second calendar to sit underneath the first calendar, and ensure they are vertically aligned.**

Use the alignment buttons on the Interactive Reporting Studio toolbar.

7 **Select the calendars, and move them onto the side-label navigation panel.**

Figure 10 Example of Calendars Dragged onto Side-Label Navigation Panel



			Revenue
Americas	North America	Canada	$5,711,432
		USA	$61,254,996
	South America	Argentina	$692,300
		Brazil	$1,097,108
Asia Pacific	Asia	Japan	$14,957,294
	Pacific Rim	Australia	$14,598,717
Europe	British Isles	Ireland	$3,159,783
		United Kingdom	$15,053,893
	Scandinavia	Norway	$1,627,859
		Sweden	$2,774,068
	Western	France	$5,976,122
		Germany	$5,978,108

Proceed to Configuring the Calendar Set Button.

Configuring the Calendar Set Button

Configure Set (`cbtSet_1`) to recognize the filter to which it is attached.

➤ To configure Set (`cbtSet_1`):

1 **Use the calendars from the Duplicating and Positioning Calendars procedure, and double-click Set.**

Properties is displayed.

2 **In Title, after Set, press Shift+Enter twice and enter the filter name.**

For example, enter `Sale Date`. The feature code recognizes the filter name, *Sale Date*, and attaches itself to that filter. Modify the filter name to match your data.

Proceed to Adding Code to the Set and Clear Buttons.

Adding Code to the Set and Clear Buttons

Add the code that enables Set and Clear.

➤ To add code to Set:

1 **Use the calendars from the Configuring a Calendar Set Button procedure, and right-click Set (`cbtSet_1`).**

2 **Select Scripts.**

Script Editor is displayed.

3 **In Script Editor, enter code to execute Set when the button is clicked.**

For example, enter `setCal(this)`. This example refers directly to the functions under the Adding Code for FrameActivate, Clearing, Setting, and Activation Functions procedure.

4 **Click OK.**

5 **Repeat steps 1–4 for Set (`cbtSet_2`).**

➤ To add code to Clear:

1 **Use the calendars from the Configuring a Calendar Set Button procedure, and right-click Clear (`cbtClear^Qiq_cal^1`).**

2 **Select Scripts.**

Script Editor is displayed.

3 **In Script Editor, enter code to execute Clear when the button is clicked.**

For example, enter `clearRangeCal(this)`. This example refers directly to the functions under the Adding Code for FrameActivate, Clearing, Setting, and Activation Functions procedure.

4 **Click OK.**

5 **Repeat steps 1–4 for Clear (`cbtClear^Qiq_cal^2`).**

Adding Code for FrameActivate, Clearing, Setting, and Activation Functions

The following code belongs in the FrameActivate button of 1100 Custom FrameType. Also see "1100 Custom" on page 207.

```
var trl=0
var thisSection=this.Parent
var cnCustom = ActiveDocument.Sections[BaseFrameType.Text + " Custom"]
Qiq_enter("1100()",trl,thisSection)

function getFilterName(obj){
   Qiq_enter("getFilterName() ",trl+1,thisSection)
   // extract the filter name from the text property of the button
   // the name is after the last newlile
   var arrFilterName = obj.Text.split("\r\n")
   Qiq_exit("getFilterName() " + arrFilterName[arrFilterName.length-...
...1],trl+1,thisSection)
   return arrFilterName[arrFilterName.length-1]
}

function clearRangeCal(in_obj){
   Qiq_enter("clearRangeCal() ",trl+1,thisSection)
   var strFilterName = getFilterName(in_obj.Parent.Shapes["cbtSet_1"])
   var objStaCal = ...
...in_obj.Parent.Shapes["picDatePicture^Qiq_cal^1"].Qiq_objCalDate
   var objEndCal = ...
...in_obj.Parent.Shapes["picDatePicture^Qiq_cal^2"].Qiq_objCalDate
   var objStaDate = objStaCal.Qiq_getDate()
   var objEndDate = objEndCal.Qiq_getDate()
   var objFilter = new Qiq_filter(strFilterName)
   if (!(objEndDate instanceof Date) && !(objStaDate instanceof Date)){
      Qiq_objDocFilters.Qiq_ignoreFilter(strFilterName)
   }else{
      if (objEndDate instanceof Date) {
         objFilter.Qiq_setOperator(bqLimitOperatorLessThanOrEqual)
         objFilter.Qiq_setSelectedValues(objEndDate)
      }else{
         objFilter.Qiq_setOperator(bqLimitOperatorGreaterThanOrEqual)
         objFilter.Qiq_setSelectedValues(objStaDate)
      }
      Qiq_objDocFilters.Qiq_setFilters(objFilter)
   }
   in_obj.Parent.OnActivate()
   Qiq_exit("clearRangeCal() ",trl+1,thisSection)
}
ActiveDocument.clearRangeCal = clearRangeCal
```

These examples illustrate code that is added into Script Editor for the functions:

- Clear Function
- Set Function
- Activate Function

Note:

Code examples must be customized to suit your requirements.

Clear Function

The Clear function requires that a filter is set to ignore the function if the calendars are cleared, or a filter is set to be greater than or less than a date value depending on which calendar remains set.

```
function setCal(in_obj){
   Qiq_enter("setCal() ",trl+1,thisSection)
   var strFilterName = getFilterName(in_obj)
   var objFilter = new Qiq_filter(strFilterName)
   var objStaCal = ...
...in_obj.Parent.Shapes["picDatePicture^Qiq_cal^1"].Qiq_objCalDate
```

```
   var objEndCal = ...
...in_obj.Parent.Shapes["picDatePicture^Qiq_cal^2"].Qiq_objCalDate
   var objStaDate = objStaCal.Qiq_getDate()
   var objEndDate = objEndCal.Qiq_getDate()
   if (objStaDate instanceof Date && objEndDate instanceof Date){
      objFilter.Qiq_setOperator(bqLimitOperatorBetween)
      objFilter.Qiq_setSelectedValues([objStaCal.Qiq_getDate(),...
...objEndCal.Qiq_getDate()])
   }else{
      if (objStaDate instanceof Date){
         objFilter.Qiq_setOperator(bqLimitOperatorGreaterThanOrEqual)
         objFilter.Qiq_setSelectedValues(objStaCal.Qiq_getDate())
      }else{
         if (objEndDate instanceof Date){
            objFilter.Qiq_setOperator(bqLimitOperatorLessThanOrEqual)
            objFilter.Qiq_setSelectedValues(objEndCal.Qiq_getDate())
         }else{
            objFilter.Qiq_setIgnore(true)
         }
      }
   }
   Qiq_objDocFilters.Qiq_setFilters(objFilter)
   in_obj.Parent.OnActivate()
   Qiq_exit("setCal() ",trl+1,thisSection)
}
ActiveDocument.setCal = setCal
```

Set Function

The Set function must set a date depending on the state of the calendar. Click Set, with no date selected, or select one or two dates and click Set. Depending on the setting, a range is set as >= start date or <= end date.

```
function rangecalActivate(mySection){
   Qiq_enter("rangecalActivate() ",trl+1,thisSection)
   var blnWork = true
   blnWork = blnWork && ...
...Hysl_doesShapeExist(mySection,"picDatePicture^Qiq_cal^1")
   blnWork = blnWork && ...
...Hysl_doesShapeExist(mySection,"picDatePicture^Qiq_cal^2")
   blnWork = blnWork && Hysl_doesShapeExist(mySection,"cbtSet_1")
   blnWork = blnWork && Hysl_doesShapeExist(mySection,"cbtSet_2")
   if (!blnWork) {
      Qiq_exit("rangecalActivate() all required shapes not ...
...found",trl+1,thisSection)
      return
   }
   // call the calendar constructor and reference it to the calendar ...
...UI picture
   mySection.Shapes["picDatePicture^Qiq_cal^1"].Qiq_objCalDate = ...
...new Qiq_objCal(mySection,"1")
   mySection.Shapes["picDatePicture^Qiq_cal^2"].Qiq_objCalDate = ...
...new Qiq_objCal(mySection,"2")
   var objStaCal = ...
...mySection.Shapes["picDatePicture^Qiq_cal^1"].Qiq_objCalDate
   var objEndCal = ...
...mySection.Shapes["picDatePicture^Qiq_cal^2"].Qiq_objCalDate
   var strFilter = getFilterName(mySection.cbtSet_1)
   var harValues = Qiq_objDocFilters.Qiq_getSelectedValues(...
...strFilter, true)
   var blnIgnore = Qiq_objDocFilters.Qiq_isIgnore(strFilter)
   var limOp = Qiq_objDocFilters.Qiq_getOperator(strFilter)
   var blnStaSel = false
   var blnEndSel = false
   var objStaDate = null
   var objEndDate = null
   if (!blnIgnore) {
      for (var a in harValues){
         // identify the first and last selected items
         switch (limOp){
            case bqLimitOperatorGreaterThanOrEqual:
               objStaDate = harValues[a]
               blnStaSel = true
               break
            case bqLimitOperatorLessThanOrEqual :
               objEndDate = harValues[a]
               blnEndSel = true
               break
            case bqLimitOperatorBetween :
               objStaDate == null? objStaDate = harValues[a] : ...
...objEndDate = harValues[a]
               !blnStaSel ? blnStaSel = true : blnEndSel = true
               break
            default :
         }
      }
   }
   if (Qiq_getHarLength(harValues) < 2){
      harValues = Qiq_objDocFilters.Qiq_getAvailableValues(...
...strFilter, true)
      for (var a in harValues){
         if (objStaDate == null) {
            objStaDate = harValues[a]
            if (objEndDate != null) {
               break
            }
         }
         objEndDate = harValues[a]
      }
   }
   objStaCal.Qiq_showDate(objStaDate,true,blnStaSel,blnStaSel)
   objEndCal.Qiq_showDate(objEndDate,true,blnEndSel,blnEndSel)
   mySection.Shapes["picDatePicture^Qiq_cal^1"].objDateSet = objStaDate
   mySection.Shapes["picDatePicture^Qiq_cal^2"].objDateSet = objEndDate

   Qiq_exit("rangecalActivate() ",trl+1,thisSection)
}
```

Activate Function

The Activate function is called when the section is activated or when a filter is set. The function synchronizes the state of the calendars with the state of the date object it manages.

```
function Usr_base_Frame_PostActivate(mySection){
   Qiq_enter(mySection.Name+".PostActivate()",trl+1,thisSection)
   rangecalActivate(mySection)
   Qiq_exit(mySection.Name+".PostActivate()",trl+1,thisSection)
}
cnCustom.Usr_base_Frame_PostActivate=Usr_base_Frame_PostActivate

cnAD.Qiq_trace("Out 1100 ()",trl+1,thisSection)
```

The Activate function is the name of the standard function called by the framework. A blank version is found in this function code.

These are advantages of using this code:

- Self-describing—To configure the function, enter the name into the GUI
- Reusable
- Can be updated

Glossary

access permissions A set of operations that a user can perform on a resource.

accountability map A visual, hierarchical representation of the responsibility, reporting, and dependency structure of the accountability teams (also known as critical business areas) in an organization.

active service A service whose Run Type is set to Start rather than Hold.

active user A user who is entitled to access the system.

active user/user group The user or user group identified as the current user by user preferences. Determines default user preferences, dynamic options, access, and file permissions. You can set the active user to your user name or any user group to which you belong.

adaptive states Interactive Reporting Web Client level of permission.

aggregate cell A cell comprising several cells. For example, a data cell that uses Children(Year) expands to four cells containing Quarter 1, Quarter 2, Quarter 3, and Quarter 4 data.

aggregate limit A limit placed on an aggregated request line item or aggregated metatopic item.

alias An alternative name. For example, for a more easily identifiable column descriptor you can display the alias instead of the member name.

appender A Log4j term for destination.

application (1) A software program designed to run a specific task or group of tasks such as a spreadsheet program or database management system. (2) A related set of dimensions and dimension members that are used to meet a specific set of analytical and/or reporting requirements.

artifact An individual application or repository item; for example, scripts, forms, rules files, Interactive Reporting documents, and financial reports. Also known as an object.

attribute Characteristics of a dimension member. For example, Employee dimension members may have attributes of Name, Age, or Address. Product dimension members can have several attributes, such as a size and flavor.

attribute dimension A type of dimension that enables analysis based on the attributes or qualities of dimension members.

authentication service A core service that manages one authentication system.

axis (1) A straight line that passes through a graphic used for measurement and categorization. (2) A report aspect used to arrange and relate multidimensional data, such as filters, pages, rows, and columns. For example, for a data query in Simple Basic, an axis can define columns for values for Qtr1, Qtr2, Qtr3, and Qtr4. Row data would be retrieved with totals in the following hierarchy: Market, Product.

bar chart A chart that can consist of one to 50 data sets, with any number of values assigned to each data set. Data sets are displayed as groups of corresponding bars, stacked bars, or individual bars in separate rows.

batch POV A collection of all dimensions on the user POV of every report and book in the batch. While scheduling the batch, you can set the members selected on the batch POV.

book A container that holds a group of similar Financial Reporting documents. Books may specify dimension sections or dimension changes.

book POV The dimension members for which a book is run.

bookmark A link to a reporting document or a Web site, displayed on a personal page of a user. The two types of bookmarks are My Bookmarks and image bookmarks.

bounding rectangle The required perimeter that encapsulates the Interactive Reporting document content when embedding Interactive Reporting document sections in a personal page, specified in pixels for height and width or row per page.

cache A buffer in memory that holds data temporarily.

calculation The process of aggregating data, or of running a calculation script on a database.

Catalog pane Displays a list of elements available to the active section. If Query is the active section, a list of database tables is displayed. If Pivot is the active section, a list of results columns is displayed. If Dashboard is the active section, a list of embeddable sections, graphic tools, and control tools are displayed.

categories Groupings by which data is organized. For example, Month

cause and effect map Depicts how the elements that form your corporate strategy relate and how they work together to meet your organization's strategic goals. A Cause and Effect map tab is automatically created for each Strategy map.

cell (1) The data value at the intersection of dimensions in a multidimensional database; the intersection of a row and a column in a worksheet. (2) A logical group of nodes belonging to one administrative domain.

chart A graphical representation of spreadsheet data. The visual nature expedites analysis, color-coding, and visual cues that aid comparisons.

chart template A template that defines the metrics to display in Workspace charts.

child A member with a parent above it in the database outline.

choice list A list of members that a report designer can specify for each dimension when defining the report's point of view. A user who wants to change the point of view for a dimension that uses a choice list can select only the members specified in that defined member list or those members that meet the criteria defined in the function for the dynamic list.

clustered bar charts Charts in which categories are viewed side-by-side; useful for side-by-side category analysis; used only with vertical bar charts.

column A vertical display of information in a grid or table. A column can contain data from one field, derived data from a calculation, or textual information.

computed item A virtual column (as opposed to a column that is physically stored in the database or cube) that can be calculated by the database during a query, or by Oracle's Hyperion® Interactive Reporting Studio in the Results section. Computed items are calculations of data based on functions, data items, and operators provided in the dialog box and can be included in reports or reused to calculate other data.

connection file *See Interactive Reporting connection file (.oce).*

content Information stored in the repository for any type of file.

cookie A segment of data placed on your computer by a Web site.

correlated subqueries Subqueries that are evaluated once for every row in the parent query; created by joining a topic item in the subquery with a topic in the parent query.

critical business area (CBA) An individual or a group organized into a division, region, plant, cost center, profit center, project team, or process; also called accountability team or business area.

critical success factor (CSF) A capability that must be established and sustained to achieve a strategic objective; owned by a strategic objective or a critical process and is a parent to one or more actions.

cube A block of data that contains three or more dimensions. An Essbase database is a cube.

custom calendar Any calendar created by an administrator.

custom report A complex report from the Design Report module, composed of any combination of components.

dashboard A collection of metrics and indicators that provide an interactive summary of your business. Dashboards enable you to build and deploy analytic applications.

data function That computes aggregate values, including averages, maximums, counts, and other statistics, that summarize groupings of data.

data layout The data layout interface is used to edit a query, arrange dimensions, make alternative dimension member selections, or specify query options for the current section or data object.

data model A representation of a subset of database tables.

database connection File that stores definitions and properties used to connect to data sources and enables database references to be portable and widely used.

descendant Any member below a parent in the database outline. In a dimension that includes years, quarters, and months, the members Qtr2 and April are descendants of the member Year.

Design Report An interface in Web Analysis Studio for designing custom reports, from a library of components.

detail chart A chart that provides the detailed information that you see in a Summary chart. Detail charts appear in the Investigate Section in columns below the Summary charts. If the Summary chart shows a Pie chart, then the Detail charts below represent each piece of the pie.

dimension A data category used to organize business data for retrieval and preservation of values. Dimensions usually contain hierarchies of related members grouped within them. For example, a Year dimension often includes members for each time period, such as quarters and months.

dimension tab In the Pivot section, the tab that enables you to pivot data between rows and columns.

dimension table (1) A table that includes numerous attributes about a specific business process. (2) In Essbase Integration Services, a container in the OLAP model for one or more relational tables that define a potential dimension in Essbase.

display type One of three Web Analysis formats saved to the repository: spreadsheet, chart, and pinboard.

dog-ear The flipped page corner in the upper right corner of the chart header area.

drill-down Navigation through the query result set using the dimensional hierarchy. Drilling down moves the user perspective from aggregated data to detail. For example, drilling down can reveal hierarchical relationships between years and quarters or quarters and months.

drill-through The navigation from a value in one data source to corresponding data in another source.

dynamic report A report containing data that is updated when you run the report.

Edit Data An interface for changing values and sending edits to Essbase.

employee A user responsible for, or associated with, specific business objects. Employees need not work for an organization; for example, they can be consultants. Employees must be associated with user accounts for authorization purposes.

ending period A period enabling you to adjust the date range in a chart. For example, an ending period of "month", produces a chart showing information through the end of the current month.

exceptions Values that satisfy predefined conditions. You can define formatting indicators or notify subscribing users when exceptions are generated.

external authentication Logging on to Oracle's applications with user information stored outside the applications, typically in a corporate directory such as MSAD or NTLM.

externally triggered events Non-time-based events for scheduling job runs.

Extract, Transform, and Load (ETL) Data source-specific programs for extracting data and migrating it to applications.

fact table The central table in a star join schema, characterized by a foreign key and elements drawn from a dimension table. This table typically contains numeric data that can be related to all other tables in the schema.

filter A constraint on data sets that restricts values to specific criteria; for example, to exclude certain tables, metadata, or values, or to control access.

folder A file containing other files for the purpose of structuring a hierarchy.

footer Text or images at the bottom of report pages, containing dynamic functions or static text such as page numbers, dates, logos, titles or file names, and author names.

format Visual characteristics of documents or report objects.

free-form grid An object for presenting, entering, and integrating data from different sources for dynamic calculations.

generic jobs Non-SQR Production Reporting or non-Interactive Reporting jobs.

grid POV A means for specifying dimension members on a grid without placing dimensions in rows, columns, or page intersections. A report designer can set POV values at the grid level, preventing user POVs from affecting the grid. If a dimension has one grid value, you put the dimension into the grid POV instead of the row, column, or page.

group A container for assigning similar access permissions to multiple users.

highlighting Depending on your configuration, chart cells or ZoomChart details may be highlighted, indicating value status: red (bad), yellow (warning), or green (good).

host A server on which applications and services are installed.

host properties Properties pertaining to a host, or if the host has multiple Install_Homes, to an Install_Home. The host properties are configured from the LSC.

hyperlink A link to a file, Web page, or an intranet HTML page.

Hypertext Markup Language (HTML) A programming language specifying how Web browsers display data.

image bookmarks Graphic links to Web pages or repository items.

implied share A member with one or more children, but only one is consolidated, so the parent and child share a value.

inactive group A group for which an administrator has deactivated system access.

inactive service A service suspended from operating.

inactive user A user whose account has been deactivated by an administrator.

Install_Home A variable for the directory where Oracle's applications are installed. Refers to one instance of Oracle's application when multiple applications are installed on the same computer.

Interactive Reporting connection file (.oce) Files encapsulating database connection information, including: the database API (ODBC, SQL*Net, etc.), database software, the database server network address, and database user name. Administrators create and publish Interactive Reporting connection files (.oce).

intersection A unit of data representing the intersection of dimensions in a multidimensional database; also, a worksheet cell.

Investigation *See drill-through.*

Java Database Connectivity (JDBC) A client-server communication protocol used by Java based clients and relational databases. The JDBC interface provides a call-level API for SQL-based database access.

job output Files or reports produced from running a job.

job parameters Reusable, named job parameters that are accessible only to the user who created them.

jobs Documents with special properties that can be launched to generate output. A job can contain Interactive Reporting, SQR Production Reporting, or generic documents.

join A link between two relational database tables or topics based on common content in a column or row. A join typically occurs between identical or similar items within different tables or topics. For example, a record in the Customer table is joined to a record in the Orders table because the Customer ID value is the same in each table.

JSP Java Server Pages.

layer (1) The horizontal location of members in a hierarchical structure, specified by generation (top down) or level (bottom up). (2) Position of objects relative to other objects. For example, in the Sample Basic database, Qtr1 and Qtr4 are in the same layer, so they are also in the same generation, but in a database with a ragged hierarchy, Qtr1 and Qtr4 might not be in same layer, though they are in the same generation.

legend box A box containing labels that identify the data categories of a dimension.

level A layer in a hierarchical tree structure that defines database member relationships. Levels are ordered from the bottom dimension member (level 0) up to the parent members.

line chart A chart that displays one to 50 data sets, each represented by a line. A line chart can display each line stacked on the preceding ones, as represented by an absolute value or a percent.

link (1) A reference to a repository object. Links can reference folders, files, shortcuts, and other links. (2) In a task flow, the point where the activity in one stage ends and another begins.

linked data model Documents that are linked to a master copy in a repository

linked reporting object (LRO) A cell-based link to an external file such as cell notes, URLs, or files with text, audio, video, or pictures. (Only cell notes are supported for Essbase LROs in Financial Reporting.)

local report object A report object that is not linked to a Financial Reporting report object in Explorer. *Contrast with linked reporting object (LRO).*

local results A data model's query results. Results can be used in local joins by dragging them into the data model. Local results are displayed in the catalog when requested.

locked data model Data models that cannot be modified by a user.

LSC services Services configured with the Local Service Configurator. They include Global Services Manager (GSM), Local Services Manager (LSM), Session Manager, Authentication Service, Authorization Service, Publisher Service, and sometimes, Data Access Service (DAS) and Interactive Reporting Service.

Map Navigator A feature that displays your current position on a Strategy, Accountability, or Cause and Effect map, indicated by a red outline.

master data model An independent data model that is referenced as a source by multiple queries. When used, “Locked Data Model” is displayed in the Query section's Content pane; the data model is linked to the master data model displayed in the Data Model section, which an administrator may hide.

MDX (multidimensional expression) The language that give instructions to OLE DB for OLAP- compliant databases, as SQL is used for relational databases. When you build the OLAPQuery section's Outliner, Interactive Reporting Clients translate requests into MDX instructions. When you process the query, MDX is sent to the database server, which returns records that answer your query. *See also SQL spreadsheet.*

measures Numeric values in an OLAP database cube that are available for analysis. Measures are margin, cost of goods sold, unit sales, budget amount, and so on. *See also fact table.*

member A discrete component within a dimension. A member identifies and differentiates the organization of similar units. For example, a time dimension might include such members as Jan, Feb, and Qtr1.

member list A named group, system- or user-defined, that references members, functions, or member lists within a dimension.

metadata A set of data that defines and describes the properties and attributes of the data stored in a database or used by an application. Examples of metadata are dimension names, member names, properties, time periods, and security.

metric A numeric measurement computed from business data to help assess business performance and analyze company trends.

MIME Type (Multipurpose Internet Mail Extension) An attribute that describes the data format of an item, so that the system knows which application should open the object. A file's mime type is determined by the file extension or HTTP header. Plug-ins tell browsers what mime types they support and what file extensions correspond to each mime type.

minireport A report component that includes layout, content, hyperlinks, and the query or queries to load the report. Each report can include one or more minireports.

missing data (#MISSING) A marker indicating that data in the labeled location does not exist, contains no value, or was never entered or loaded. For example, missing data exists when an account contains data for a previous or future period but not for the current period.

model (1) In data mining, a collection of an algorithm's findings about examined data. A model can be applied against a wider data set to generate useful information about that data. (2) A file or content string containing an application-specific representation of data. Models are the basic data managed by Shared Services, of two major types: dimensional and non-dimensional application objects. (3) In Business Modeling, a network of boxes connected to represent and calculate the operational and financial flow through the area being examined.

multidimensional database A method of organizing, storing, and referencing data through three or more dimensions. An individual value is the intersection point for a set of dimensions.

native authentication The process of authenticating a user name and password from within the server or application.

note Additional information associated with a box, measure, scorecard or map element.

null value A value that is absent of data. Null values are not equal to zero.

online analytical processing (OLAP) A multidimensional, multiuser, client-server computing environment for users who analyze consolidated enterprise data in real time. OLAP systems feature drill-down, data pivoting, complex calculations, trend analysis, and modeling.

origin The intersection of two axes.

page member A member that determines the page axis.

palette A JASC compliant file with a .PAL extension. Each palette contains 16 colors that complement each other and can be used to set the dashboard color elements.

performance indicator An image file used to represent measure and scorecard performance based on a range you specify; also called a status symbol. You can use the default performance indicators or create an unlimited number of your own.

personal pages A personal window to repository information. You select what information to display and its layout and colors.

personal recurring time events Reusable time events that are accessible only to the user who created them.

personal variable A named selection statement of complex member selections.

perspective A category used to group measures on a scorecard or strategic objectives within an application. A perspective can represent a key stakeholder (such as a customer, employee, or shareholder/financial) or a key competency area (such as time, cost, or quality).

pie chart A chart that shows one data set segmented in a pie formation.

pinboard One of the three data object display types. Pinboards are graphics, composed of backgrounds and interactive icons called pins. Pinboards require traffic lighting definitions.

pins Interactive icons placed on graphic reports called pinboards. Pins are dynamic. They can change images and traffic lighting color based on the underlying data values and analysis tools criteria.

plot area The area bounded by X, Y, and Z axes; for pie charts, the rectangular area surrounding the pie.

predefined drill paths Paths used to drill to the next level of detail, as defined in the data model.

presentation A playlist of Web Analysis documents, enabling reports to be grouped, organized, ordered, distributed, and reviewed. Includes pointers referencing reports in the repository.

primary measure A high-priority measure important to your company and business needs. Displayed in the Contents frame.

Production Reporting *See SQR Production Reporting*.

promotion The process of transferring artifacts from one environment or machine to another; for example, from a testing environment to a production environment.

property A characteristic of an artifact, such as size, type, or processing instructions.

proxy server A server acting as an intermediary between workstation users and the Internet to ensure security.

public job parameters Reusable, named job parameters created by administrators and accessible to users with requisite access privileges.

public recurring time events Reusable time events created by administrators and accessible through the access control system.

publish The process that enables a model owner to forward a model or model changes for inclusion in an enterprise model.

range A set of values including upper and lower limits, and values falling between limits. Can contain numbers, amounts, or dates.

reconfigure URL URL used to reload servlet configuration settings dynamically when users are already logged on to the Workspace.

recurring time event An event specifying a starting point and the frequency for running a job.

relational database A type of database that stores data in related two-dimensional tables. *Contrast with multidimensional database*.

report object In report designs, a basic element with properties defining behavior or appearance, such as text boxes, grids, images, and charts.

resources Objects or services managed by the system, such as roles, users, groups, files, and jobs.

result frequency The algorithm used to create a set of dates to collect and display results.

role The means by which access permissions are granted to users and groups for resources.

row heading A report heading that lists members down a report page. The members are listed under their respective row names.

RSC services Services that are configured with Remote Service Configurator, including Repository Service, Service Broker, Name Service, Event Service, and Job Service.

scale The range of values on the Y axis of a chart.

schedule Specify the job that you want to run and the time and job parameter list for running the job.

score The level at which targets are achieved, usually expressed as a percentage of the target.

scorecard Business Object that represents the progress of an employee, strategy element, or accountability element toward goals. Scorecards ascertain this progress based on data collected for each measure and child scorecard added to the scorecard.

scorecard report A report that presents the results and detailed information about scorecards attached to employees, strategy elements, and accountability elements.

secondary measure A low-priority measure, less important than primary measures. Secondary measures do not have Performance reports but can be used on scorecards and to create dimension measure templates.

Section pane Lists all sections that are available in the current Interactive Reporting Client document.

security agent A Web access management provider (for example, Netegrity SiteMinder) that protects corporate Web resources.

security platform A framework enabling Oracle's applications to use external authentication and single sign-on.

services Resources that enable business items to be retrieved, changed, added, or deleted. Examples: Authorization and Authentication.

servlet A piece of compiled code executable by a Web server.

Servlet Configurator A utility for configuring all locally installed servlets.

sibling A child member at the same generation as another child member and having the same immediate parent. For example, the members Florida and New York are children of East and each other's siblings.

single sign-on Ability to access multiple Oracle's products after a single login using external credentials.

SmartCut A link to a repository item, in URL form.

snapshot Read-only data from a specific time.

SPF files Printer-independent files created by a SQR Production Reporting server, containing a representation of the actual formatted report output, including fonts, spacing, headers, footers, and so on.

Spotlighter A tool that enables color coding based on selected conditions.

SQL spreadsheet A data object that displays the result set of a SQL query.

SQR Production Reporting A specialized programming language for data access, data manipulation, and creating SQR Production Reporting documents.

stacked charts A chart where the categories are viewed on top of one another for visual comparison. This type of chart is useful for subcategorizing within the current category. Stacking can be used from the Y and Z axis in all chart types except pie and line. When stacking charts the Z axis is used as the Fact/Values axis.

Start in Play The quickest method for creating a Web Analysis document. The Start in Play process requires you to specify a database connection, then assumes the use of a spreadsheet data object. Start in Play uses the highest aggregate members of the time and measures dimensions to automatically populate the rows and columns axes of the spreadsheet.

strategic objective (SO) A long-term goal defined by measurable results. Each strategic objective is associated with one perspective in the application, has one parent, the entity, and is a parent to critical success factors or other strategic objectives.

Strategy map Represents how the organization implements high-level mission and vision statements into lower-level, constituent strategic goals and objectives.

structure view Displays a topic as a simple list of component data items.

Structured Query Language A language used to process instructions to relational databases.

subscribe Flags an item or folder to receive automatic notification whenever the item or folder is updated.

Summary chart In the Investigates Section, rolls up detail charts shown below in the same column, plotting metrics at the summary level at the top of each chart column.

super service A special service used by the startCommonServices script to start the RSC services.

target Expected results of a measure for a specified period of time (day, quarter, etc.,)

time events Triggers for execution of jobs.

time scale Displays metrics by a specific period in time, such as monthly or quarterly.

token An encrypted identification of one valid user or group on an external authentication system.

top and side labels Column and row headings on the top and sides of a Pivot report.

top-level member A dimension member at the top of the tree in a dimension outline hierarchy, or the first member of the dimension in sort order if there is no hierarchical relationship among dimension members. The top-level member name is generally the same as the dimension name if a hierarchical relationship exists.

trace level Defines the level of detail captured in the log file.

traffic lighting Color-coding of report cells, or pins based on a comparison of two dimension members, or on fixed limits.

transformation (1) Transforms artifacts so that they function properly in the destination environment after application migration. (2) In data mining, modifies data (bidirectionally) flowing between the cells in the cube and the algorithm.

transparent login Logs in authenticated users without launching the login screen.

trusted password A password that enables users authenticated for one product to access other products without reentering their passwords.

trusted user Authenticated user

user directory A centralized location for user and group information. Also known as a repository or provider.

Web server Software or hardware hosting intranet or Internet Web pages or Web applications.

weight Value assigned to an item on a scorecard that indicates the relative importance of that item in the calculation of the overall scorecard score. The weighting of all items on a scorecard accumulates to 100%. For example, to recognize the importance of developing new features for a product, the measure for New Features Coded on a developer's scorecard would be assigned a higher weighting than a measure for Number of Minor Defect Fixes.

ws.conf A configuration file for Windows platforms.

wsconf_platform A configuration file for UNIX platforms.

Y axis scale Range of values on Y axis of charts displayed in Investigate Section. For example, use a unique Y axis scale for each chart, the same Y axis scale for all Detail charts, or the same Y axis scale for all charts in the column. Often, using a common Y axis improves your ability to compare charts at a glance.

Zero Administration Software tool that identifies version number of the most up-to-date plug-in on the server.

zoom Sets the magnification of a report. For example, magnify a report to fit whole page, page width, or percentage of magnification based on 100%.

ZoomChart Used to view detailed information by enlarging a chart. Enables you to see detailed numeric information on the metric that is displayed in the chart.

Index