Quest

Foglight® 8.3.0

Web Component Tutorial

© 2026 Quest Software Inc.

ALL RIGHTS RESERVED.

This guide contains proprietary information protected by copyright. The software described in this guide is furnished under a software license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of the applicable agreement. No part of this guide may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of Quest Software Inc.

The information in this document is provided in connection with Quest Software products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Quest Software products. EXCEPT AS SET FORTH IN THE TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, QUEST SOFTWARE ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL QUEST SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF QUEST SOFTWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Quest Software makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Quest Software does not make any commitment to update the information contained in this document.

If you have any questions regarding your potential use of this material, contact:

Quest Software Inc.
Attn: LEGAL Dept.
4 Polaris Way
Aliso Viejo, CA 92656

Refer to our website (https://www.quest.com) for regional and international office information.

Patents

Quest Software is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at https://www.quest.com/legal.

Trademarks

Quest, the Quest logo, and Join the Innovation are trademarks and registered trademarks of Quest Software Inc. For a complete list of Quest marks, visit https://www.quest.com/legal/trademark-information.aspx. "Apache HTTP Server", Apache, "Apache Tomcat" and "Tomcat" are trademarks of the Apache Software Foundation. Google is a registered trademark of Google Inc. Android, Chrome, Google Play, and Nexus are trademarks of Google Inc. Red Hat, JBoss, the JBoss logo, and Red Hat Enterprise Linux are registered trademarks of Red Hat, Inc. in the U.S. and other countries. CentOS is a trademark of Red Hat, Inc. in the U.S. and other countries. Fedora and the Infinity design logo are trademarks of Red Hat, Inc. Microsoft, .NET, Active Directory, Internet Explorer, Hyper-V, Office 365, SharePoint, Silverlight, SQL Server, Visual Basic, Windows, Windows Vista and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. AIX, IBM, PowerPC, PowerVM, and WebSphere are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Java, Oracle, Oracle Solaris, PeopleSoft, Siebel, Sun, WebLogic, and ZFS are trademarks or registered trademarks of Oracle and/or its affiliates in the United States and other countries. SPARC is a registered trademark of SPARC International, Inc. in the United States and other countries. Products bearing the SPARC trademarks are based on an architecture developed by Oracle Corporation. OpenLDAP is a registered trademark of the OpenLDAP Foundation. HP is a registered trademark that belongs to Hewlett-Packard Development Company, L.P. Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both. MySQL is a registered trademark of MySQL AB in the United States, the European Union and other countries. Novell and eDirectory are registered trademarks of Novell, Inc., in the United States and other countries. VMware, ESX, ESXi, vSphere, vCenter, vMotion, and vCloud Director are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions. Sybase is a registered trademark of Sybase, Inc. The X Window System and UNIX are registered trademarks of The Open Group. Mozilla and Firefox are registered trademarks of the Mozilla Foundation. "Eclipse", "Eclipse Foundation Member", "EclipseCon", "Eclipse Summit", "Built on Eclipse", "Eclipse Ready" "Eclipse Incubation", and "Eclipse Proposals" are trademarks of Eclipse Foundation, Inc. IOS is a registered trademark or trademark of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries. Apple, iPad, iPhone, Mac OS, Safari, Swift, and Xcode are trademarks of Apple Inc., registered in the U.S. and other countries. Ubuntu is a registered trademark of Canonical Ltd. Symantec and Veritas are trademarks or registered trademarks of Symantec Corporation or its affiliates in the U.S. and other countries. OpenSUSE, SUSE, and YAST are registered trademarks of SUSE LCC in the United States and other countries. Citrix, AppFlow, NetScaler, XenApp, and XenDesktop are trademarks of Citrix Systems, Inc. and/or one or more of its subsidiaries, and may be registered in the United States Patent and Trademark Office and in other countries. AlertSite and DéjàClick are either trademarks or registered trademarks of Boca Internet Technologies, Inc. Samsung, Galaxy S, and Galaxy Note are registered trademarks of Samsung Electronics America, Inc. and/or its related entities. MOTOROLA is a registered trademarks of Motorola Trademark Holdings, LLC. The Trademark BlackBerry Bold is owned by Research In Motion Limited and is registered in the United States and may be pending or registered in other countries. Quest is not endorsed, sponsored, affiliated with or otherwise authorized by Research In Motion Limited. Ixia and the Ixia four-petal logo are registered trademarks or trademarks of Ixia. Opera, Opera Mini, and the O logo are trademarks of Opera Software ASA. Tevron, the Tevron logo, and CitraTest are registered trademarks of Tevron, LLC. PostgreSQL is a registered trademark of the PostgreSQL Global Development Group. MariaDB is a trademark or registered trademark of MariaDB Corporation Ab in the European Union and United States of America and/or other countries. Vormetric is a registered trademark of Vormetric, Inc. Intel, Itanium, Pentium, and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries. Debian is a registered trademark of Software in the Public Interest, Inc. OpenStack is a trademark of the OpenStack Foundation. Amazon Web Services, the "Powered by Amazon Web Services" logo, and "Amazon RDS" are trademarks of Amazon.com, Inc. or its affiliates in the United States and/or other countries. Infobright, Infobright Community Edition and Infobright Enterprise Edition are trademarks of Infobright Inc. POLYCOM®, RealPresence® Collaboration Server, and RMX® are registered trademarks of Polycom, Inc. All other trademarks and registered trademarks are property of their respective

owners.

Legend

WARNING: A WARNING icon indicates a potential for property damage, personal injury, or death.

CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.

IMPORTANT NOTE, **NOTE**, **TIP**, **MOBILE**, or **VIDEO:** An information icon indicates supporting information.

Foglight Web Component
Tutorial Foglight Version - 8.3.0

Contents

Using the Web Component Tutorial . . . 7
Browser Interface Views . . . 7
Why Configure the Default Views? . . . 7
What is the Web Component Framework? . . . 8
Simple Dashboard Creation . . . 10
How to Use this Tutorial . . . 10

Tutorial 1: Drag and Drop . . . 12
Creating a Custom Dashboard by Drag and Drop . . . 12
Customizing Dashboards . . . 17
Add View . . . 17
Add Text . . . 18
Delete This Page . . . 19
Edit Page Layout . . . 21

Tutorial 2: Creating a Dashboard . . . 31
Constructing the Tutorial Module . . . 32
Configuring a Query . . . 33
Configuring a Row-Oriented Table . . . 35
Configuring Table Columns . . . 36
Building the Dashboard . . . 37
Adding the Table to the Dashboard . . . 39
Summary . . . 40

Tutorial 3: Adding a Drilldown Page . . . 42
Preliminary Setup . . . 42
Adding a Column of Aggregate Alarms to the Table . . . 43
Adding a Dependent Page . . . 45
Adding a Context-Sensitive Dependent Page for the Table of Hosts . . . 46
Summary . . . 49

Tutorial 4: Adding Views . . . 52
Designing the Page . . . 53
Preliminary Setup . . . 53
Defining the Alarms Query . . . 54
Configuring a Table of Alarms . . . 55
Configuring a Chart Component . . . 59
Configuring a Label with an Action . . . 61
Configuring the Drilldown Page . . . 63
Configuring a Drop-Down List . . . 64
Flowing Monitored Hosts to Host Details T3 . . . 66
Adding a Customizer . . . 66
Adding Nested Views . . . 67
Summary . . . 68

Tutorial 5: Using a Grid 70
Building the Dashboard 72
Summary 78

Tutorial 6: Reports 80
Configuring a Query 81
Creating a Basic Report Page 81
Creating a More Elaborate Hosts Table 84
Adding a Header 89
Adding a Footer 90
Adding an Iterator for a Multi-Page Report 92
Choosing the Iterated View 93
Summary 93

Tutorial 7: Creating a Form 97
About this Tutorial 97
Defining the Enumerated Types for Manager and Staff 100
Defining the Manager and Staff Types 101
Defining the Company Type 102
Defining the Functions for Adding and Removing a Manager 103
Defining the Labels 104
Defining the Bean Input Components 104
Defining the Grid Layout 106
Defining the Dialog 108
Defining the Table of Managers 109
Defining the Top-Level Dashboard 112
Summary 113

Tutorial 8: Renderers 116
Text Renderer 118
Date Renderer 119
Number Bar Renderer and Number Unit Renderer 120
Configuring the Labels 121

Tutorial 9: Adding Questions and Answers 123
Working with Associations and Question Viewers 123
Building Associations for Your Question Viewer 125
Creating a Domain 126
Creating Categories 127
Creating Questions 127
Populating the Question Viewer with Questions and Answers 129
Building a Mini-Viewer 133
Summary 135

Tutorial 10: Sending Messages to Other Users 136
Creating Message Associations 136
Creating Script Functions 137

Creating Label Views 138
Testing Messages 139
Summary 142

About Us 143
We are more than just a name 143
Our brand, our vision. Together. 143
Contacting Quest 143
Technical support resources 143

1

Using the Web Component Tutorial

With Foglight®, application and IT managers can understand end-user service levels for their critical business applications, notify stakeholders when those service levels are violated, and assign problem resolution tasks to the appropriate domain expert.

In addition to Foglight's core performance management capabilities, Foglight offers specialized monitoring for all application tiers including: End-User Response, Application, Web Servers, Application Servers, Databases, and Operating Systems.

That is a lot of capability, and the volume of data being collected can be overwhelming. The views in the browser interface attempt to organize the data into meaningful summaries, with drilldowns to increasingly specific information about a chosen component, such as a single host or a particular database instance.

It is likely that you organize the top-level screens around the concept of services, and choose them to show a view that should be useful to a broad range of users—those with typical environments. In all likelihood your environment is not quite typical, and as you gain familiarity with the browser interface, you will imagine ways that they could be improved.

NOTE: You must have a Foglight Dashboard Designer role to develop new dashboards.

- How to Use this Tutorial

Browser Interface Views

Web Component designers anticipated your need to customize the browser interface, so they included the means to allow you to access the product's component framework and create your own custom views. You can populate these views with other display components, such as charts and tables, and connect them to data sources. This is the same data that the Foglight® agents are configured to collect, but now it is organized in a way that best fits your business model and its information needs.

The end result is a monitoring system that organizes data in a way that mirrors your business model. Real-time monitoring data is presented the way you want to see it, and you better control your application's availability. This also helps you with service level management: because your custom views show services in a cleaner way, you can inform application and IT managers about end-user service levels, notify stakeholders when those service levels are violated, and assign problem resolution tasks to the appropriate domain experts. Custom views that focus on known trouble spots can help establish processes for quick recovery from system failure.

Why Configure the Default Views?

Suppose your company's rapid expansion leads to the addition of many different application systems and groups, with each support team comfortable with their own legacy systems. In total, your IT department is responsible for managing an ERP system consisting of many servers and Oracle® databases distributed across a number of major locations.

Before the acquisition of Foglight®, homegrown scripts and applications were used to monitor these distributed systems, which made it difficult for the support organization to manage the information and care for the entire distributed environment.

Now you have Foglight, a standardized monitoring system that provides centralized management and allows people to attack problems from the same perspective, pro-actively monitor a large heterogeneous environment, and offers access to this distributed monitoring system via a Web browser. You and your team can see the big picture and correlate events between systems for the entire application. With Foglight, the situation can be improved even further.

Foglight uses a configurable Web-based interface. By making your own custom configurations, you can apply your detailed knowledge of your system to augment or replace the views Foglight shows by default.

This helps Foglight to notify you directly when something is broken, rather than relying on a user notifying a help desk, getting a trouble ticket, and only then having the support group notified. If a custom configuration presents a clearer view or better correlation of events between all systems, and leads to pinpointing a problem that might otherwise take longer to notice and to diagnose, then the time spent in crafting the custom view is well spent.

The aim of this tutorial is to make the learning curve less steep. In it, you'll see how to create additional views, populate them with GUI components, and connect these components to data sources.

What is the Web Component Framework?

The Web Component Framework is a super-set of the View Component collection that contains other control components, such as renderers. It is used to build thin client interfaces for products that are primarily (but not necessarily) in the systems management domain. This is the framework you will use in this tutorial.

The Web Component Framework is written in Java™ and is capable of running in a Web container such as Apache Tomcat. It can be used on contemporary Web browsers without requiring the use of a plug-in. It is portal-like, but is not a JSR-168 standard portal.

The top level of the Web Component Framework is comprised of panels.The items in the left and right panels can provide access to several different display types:

- A context-free view, which can be a dashboard or portal
- A specialized browser, which is a two-pane display—navigator and page
- A page, which is a special high-level construct that contains one or more views for which a context is required, and is normally used for a drilldown view
- A custom display in which the application supplies an HTML fragment

View components

View components are the visible components in the user interface. Multiple components can be arranged on a page and some components can be nested within others. A view contains both view components and configuration information.

Types of view components:

- *Containers*, such as various layouts, splitters, report generator
- *Data visualization components*, such as charts, tables, gauges, labels, trees
- *Specialized components*, such as RSS feeds

The configuration information includes flow control, contextual inputs, data binding, and query specification.

There are two styles to choose from, a browser, which contains a navigator, or the plain view that presents a simple page without a navigation component.

Pages can be decorated with headers, which may contain:

- *Breadcrumbs*, a page name preceded by other page names, representing the navigation path to the current page
- *Time region*, which may contain:
    - *Timestamp*: if no time range is applied to the page

    - *Time Range*: if available, applies to all views
    - *Nothing*: if multiple time ranges are represented
- *Page scope actions*, such as:
    - *View menu*: Actions you can perform on the view
    - *Help menu*: Help contents for all components on the current page
- Optional page scope actions, such as:
    - *Time Range*: Change the time range of the page
    - *View menu*: In a browser, causes a switch to a different page

View layouts

The following view layouts are available:

Grid

- Views can size themselves appropriately or be hard-coded to a fixed size.
- Views with title bars can be collapsed and expanded.
- Fixed size views grow their own scroll bars as necessary while variable sized components cause the browser to grow scroll bars as necessary.

Fixed

- All views are exactly placed and sized. This layout is useful for monitoring views with fixed real estate when you want to make sure that all the important information is always on the screen.

Iterator

- A chosen view is repeated for every item in a list. For instance, the view might be a chart of CPU utilization and the list would contain all the hosts in the monitored system.

Numerous other layouts are available. The ones listed here are discussed in this tutorial. For information on the other layouts, refer to the *Web Component Reference* help, which can be accessed from the **Help** tab located in the browser interface action panel.

NOTE: The action panel is sometimes referred to as the Page panel in the *Web Component Reference*.

Data sources

Data sources encapsulate all that the system knows about the data and yet cleanly separates knowledge of the data from how it is presented.

The data source is organized as a dynamic graph of objects, starting from a root that represents the entire data model.

Objects are defined in the API and are not tied to the creation of any particular Java object.

The implementation of the data source must provide a schema for the objects. The schema can be changed dynamically.

Schema must provide:

- Object types
- Object properties
- Property units and bounds
- Localizations for property names, object types, unit labels

Data source queries

The following principles apply to data source queries:

- Syntax resembles SQL, but queries return objects
- Strongly typed
- Not free-form

Simple Dashboard Creation

It is possible to create dashboards from existing data and views by dragging them from the action panel. This tutorial shows you how to build views from scratch by using the Web Component Framework.

To create a dashboard from the action panel:

1. From the action panel, on the **General** tab, click **Create dashboard**.
2. Complete the **Create Dashboard** wizard. For complete information, see Tutorial 1: Drag and Drop on page 12.
3. From the **Views** or **Data** tab appearing on the action panel, drag any of the objects to the display area.
4. Make changes to the page refresh rate, relevant and allowed roles, and help text if necessary.

How to Use this Tutorial

This tutorial will help you to learn how to use the Configuration dashboard to create custom views that present collected data in a way that makes the most sense to you. It is aimed at users who want to provide their clients with a specially tailored set of views to augment the default views.

The default views are organized around the concept of services and can be modified easily to suit any organization's perception of its data interrelationships. There may be situations that require special views—ones that demand more than the types of reorganization afforded by the Foglight® top-level edit mechanism. In this case, it becomes necessary to work with the Web Component Framework, which was used to build the UI for Foglight. This tutorial serves as a starting place for learning how to use the Web Component Reference, which describes all the components, renderers, and other elements in the Web Component Framework.

You begin by creating a very simple view, called a dashboard, that contains a table whose rows are all the hosts in your monitored space. From there, other examples illustrate how most common services in your monitored space can be reconfigured to suit your special circumstances.

The first part of the tutorial contains a discussion of the basic operations, such as creating a view that presents some static data. Further examples illustrate more complex objectives, such as creating a page that updates itself on the basis of a selection made in a drop-down list on the page itself.

New users—those unfamiliar with Foglight —should complete the first exercise to familiarize themselves with the browser interface. When you are familiar with the various editing pages and dialog boxes, you can tackle the more complex examples.

Printing the tutorial

To print topics in this tutorial, click the print icon on the browser's menu bar or print from the PDF version.

Getting more help

While this tutorial shows you how to use some of Foglight® Web Component features, you may have questions about additional components and features. You can find information about using all of the components in the Quest Web Component Reference. To access the *Web Component Reference*, expand the View Components folder in the *Web Component Reference* help. Use the table of contents in the left pane to navigate to the information you need.

2

Tutorial 1: Drag and Drop

This tutorial shows you how to create a custom dashboard using the **Create Dashboard** wizard. This wizard includes the following features:

- You can create a dashboard from existing views and data objects and have them positioned in a one-, two-, or three-column layout.
- You can create a fixed-position layout within which you can size views and position them exactly where you want them. With this feature, you can lay out your dashboard precisely the way you want it and make the most out of the real estate available on the screen.
- For a dashboard, you can set the background to a geographical map (one of your campus or your server rack, for example) and then position the views with state precisely on top of the map to represent the monitored state of those locations. This is very effective when used in conjunction with the fixed layout dashboard feature. You can use the background images that are packaged with Foglight, or your own uploaded images.

  IMPORTANT: You can set a background image for a report as well. For a report, you can set the background to be a watermark that is repeated on each page.

- For custom dashboard views that you create yourself, you can configure the popup, dwell, and drilldown behavior so that the user is taken to a view that already exists in your system or to another dashboard or view you have created.
- Optionally, make a view available for inclusion in other custom dashboards or reports.

Many of the Create Dashboard options are now wizard-driven, similar to the options for creating a report.

Also, you can create your own views based on data in the particular object that is selected.

The *Web Component Guide* describes how Foglight® organizes the data it collects. These objects (and metrics) are displayed using a special-purpose framework.

- Creating a Custom Dashboard by Drag and Drop
- Customizing Dashboards

Creating a Custom Dashboard by Drag and Drop

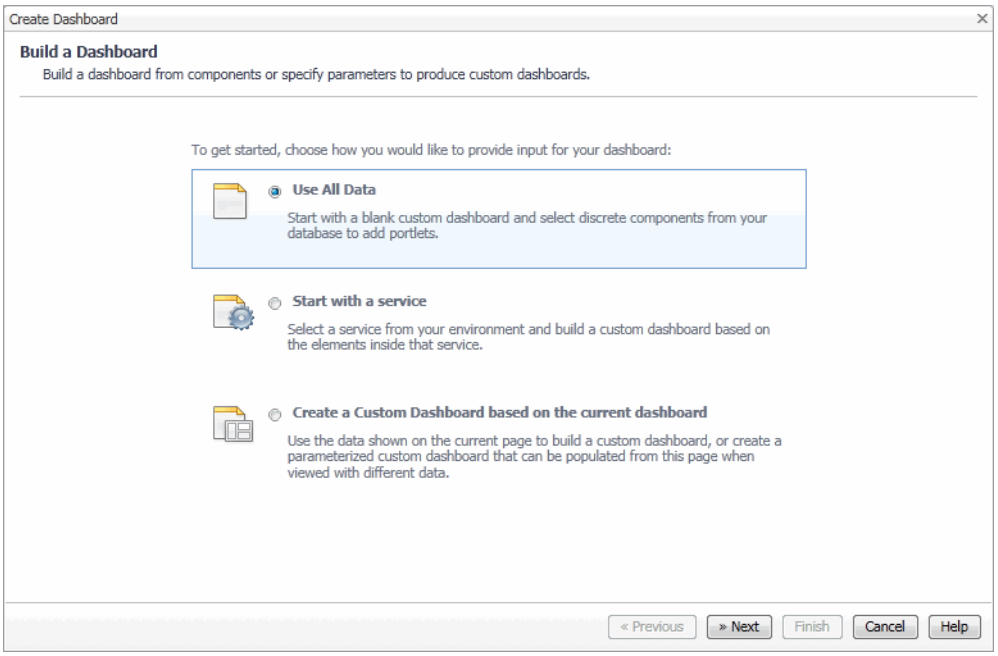
The standard Foglight® dashboards have been designed to satisfy most monitoring needs, however Foglight is also highly extensible and, if your role allows it, it gives you the capability of designing your own custom dashboards. The simplest method for creating a custom dashboard is to use the **Create Dashboard** link available on the action panel's **General** tab.

To build a dashboard from data elements:

1 On the action panel, on the **General** tab, under **Other Actions**, click **Create Dashboard**.

The **Create Dashboard** wizard appears with the **Build a Dashboard** page open.

Figure 1. Create Dashboard



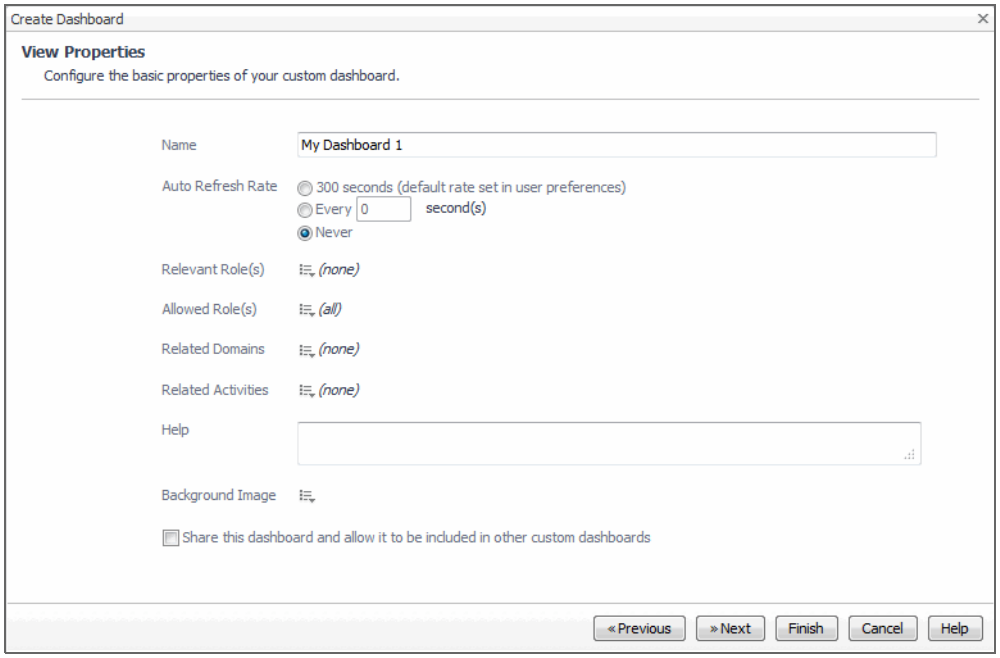
2 Choose the type of data input for your dashboard.

In the **Create Dashboard** wizard, on the **Build a Dashboard** page, select one of the following options.

- **Use All Data**: Select this option to build a dashboard from various data elements, and click **Next**.
- **Start with a Service**: Select this option to build a dashboard from service elements. On the **Select a Service** page that appears, select the service, and click **Next**.
- **Create a Custom Dashboard based on the current dashboard**: Select this option to build a dashboard from the current dashboard. On the **Select Objects** page that appears, select the data inputs from the current dashboard that you want to include in the new dashboard, and click **Next**.

3 In the **Create Dashboard** wizard, on the **View Properties** page that appears, edit the dashboard properties, as required.

Figure 2. Create Dashboard



a In the **Name** box, type the dashboard name.

b Specify the dashboard's **Auto Refresh Rate** to one of the available options:

- The default number of seconds set in **User Preferences**
- A desired number of seconds
- Never

c Select the default roles to control access to this dashboard. By default, if no dashboard roles are selected, the dashboard is available to all roles.

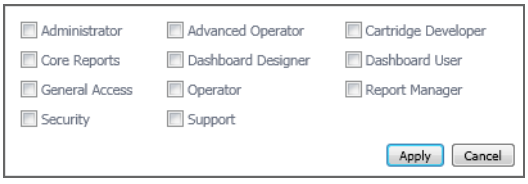
Relevant Roles control which choices appear in wizards, and Allowed Roles determine which user role is required to grant access to this dashboard.

> **TIP:** Relevant Roles are used to act as filters, showing the views a user is interested in and not showing the others. If some but not all Relevant Roles are set on a view, then in some situations the GUI may not show that view to a user who does not have that role. Even though the view does not show in, say, a list of dashboards, the user can still access the view if there is a page accessible to the user that has a link to the unlisted view.Allowed Roles act as a security mechanism. If some but not all Allowed Roles are set on a view, then only users for whom at least one of their roles is also an Allowed Role on the view can see or in any way interact with that view.

Click beside **Relevant Role(s)** and/or **Allowed Role(s)** and select the appropriate roles.

> **TIP:** Leaving the default Allowed Role(s) setting to all grants access to all roles to access the new dashboard.

Figure 3. Allowed and Relevant Roles

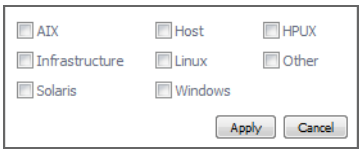


d **Optional**—Associate the monitored data with a specific domain.

Note This information is used as your reference.

Click beside **Related Domains** and select a domain in the **Select related domain** dialog box that appears.

Figure 4. Related Domains



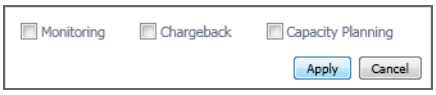
TIP: The type and range of domains that are available for selection reflects the selection of installed cartridges.

e **Optional**—Associate the monitored data with specific monitoring activities.

Note This information is used as your reference.

Click beside **Related Activities** and select one or more activities in the **Select related domain** dialog box that appears.

Figure 5. Related Activities



f Provide a brief description of the dashboard. If specified, this description appears as a popup when hovering over the dashboard name in the navigation panel.

In the **Help** box, type the dashboard description.

g Add a graphic as a background.

Click beside **Background Image** and select a graphic file.

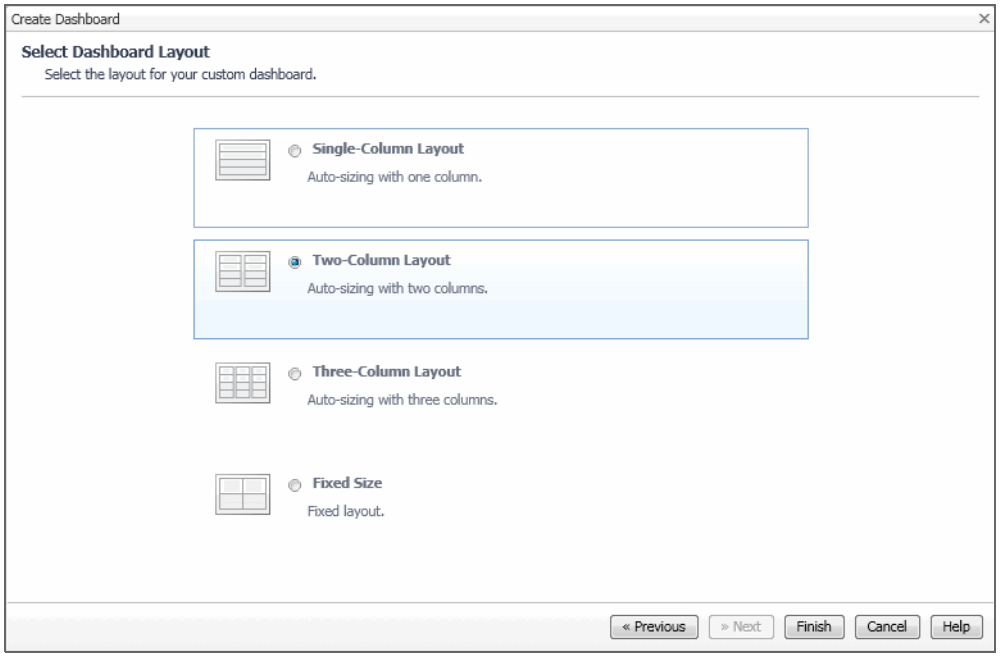
h Indicate if you want to make this dashboard to have a portlet purpose, making it available for inclusion in other custom dashboards.

Select or clear the check box **Share this dashboard and allow it to be included in other custom dashboards**.

i Click **Next**.

The **Select Dashboard Layout** page appears in the **Create Dashboard** wizard.

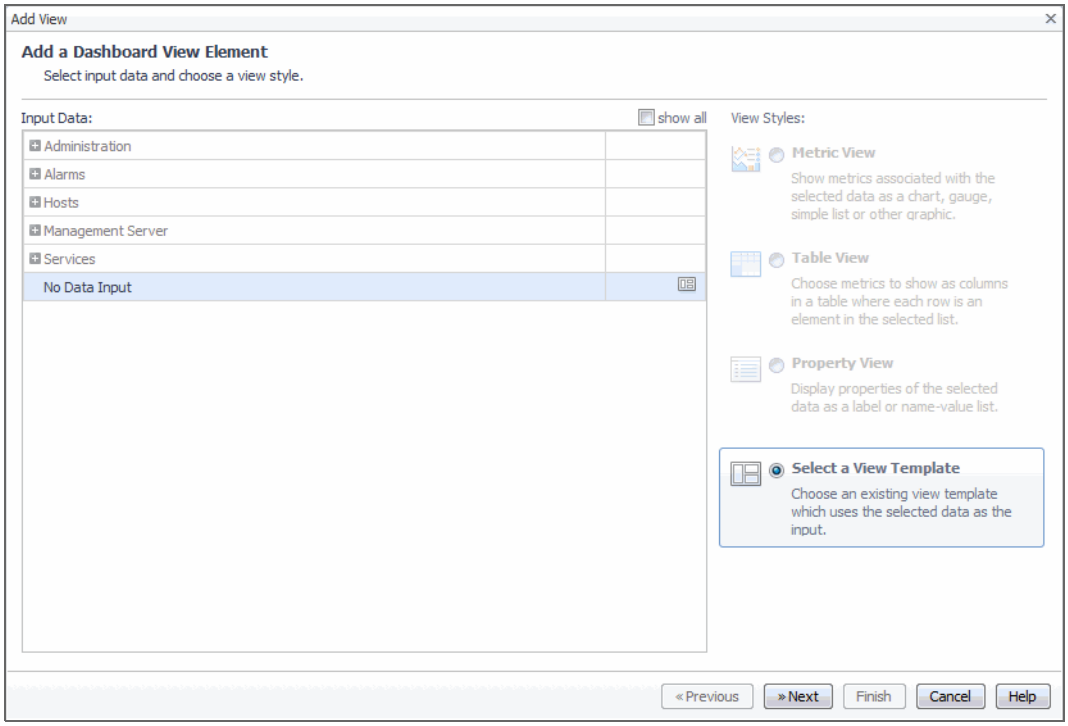
Figure 6. Select Dashboard Layout



4 Select a desired dashboard layout and click **Finish**.

The **Create Dashboard** wizard closes, and the **Add View** dialog box appears with the **Add a Dashboard View Element** page open.

Figure 7. Add View



5 Select the data that you want to use as the input and select an available view style.

a On the **Add a Dashboard View Element** page, in the **Input Data** area, expand the applicable node and select it in the tree.

For example, if you are building a dashboard from various data elements, and want to add a list of current alarms, choose **Alarms > Current Alarms**.

> **TIP:** By default, repetitive and complex properties are hidden from the Input Data area to make the selection of meaningful choices simpler. Selecting show all displays all data objects.

The **View Styles** area refreshes, reflecting your selection. Only those view templates that can accommodate the selected data type are available for selection.

b In the **View Styles** area, select an available view style.

For example, if you selected **Alarms > Current Alarms** in the **Input Data** area, you can select **Table View** or **Select a View Template**.

c Click **Next**.

The **Add View** dialog box refreshes, showing the options that are available based on the selections made in the previous step.

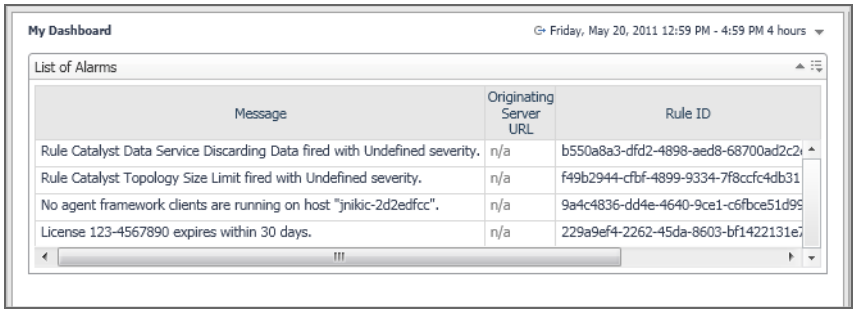
6 Complete the flow for configuring the view. The type and range of available options depends on the selected data type and the view style.

For example, if you want to want to add a tabular view that lists current alarms, you can select the columns that you want to appear in the table, the column headers, alignment, and renderers, then preview the table, and filter the data.

7 When completed, click **Finish**.

The view appears on the newly created dashboard in the display area.

Figure 8. List of Alarms



> **IMPORTANT:** When you have finished adding views to your dashboard, you do not need to save it. It is automatically available under your working module, which is My Dashboards by default.

Customizing Dashboards

Add View

To help you with adding data to your dashboard, several functions are available:

- Add View (see page 17)
- Add Text (see page 18)
- Delete This Page (see page 19)
- Edit Page Layout (see page 21)

This topic describes the process of using the **Add View** wizard. This wizard is provided for the same function needed when creating a custom report.

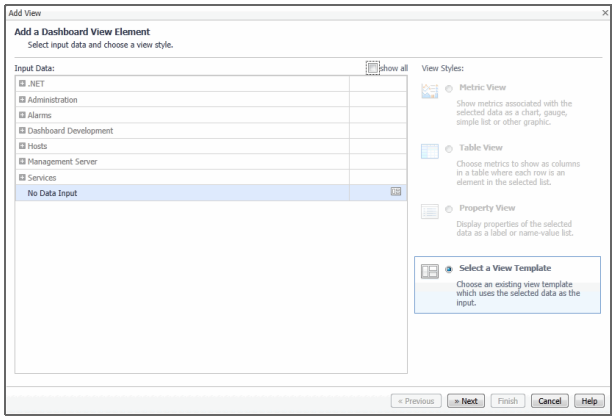
To add more views using the Add View wizard:

1 If the dashboard is not already opened, select the dashboard from the **My Dashboards** node.

2 Click **Add view** from the **Action** panel.

The **Add View** wizard appears.

Figure 9. Add View



3 Continue using the procedure for adding view elements.

Add Text

You can use the Add Text feature to add a description of the dashboard, change the font type, or add a background image.

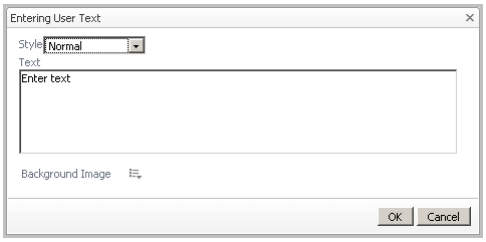
To add text to the dashboard:

1 If the dashboard is not already opened, select the dashboard from the **My Dashboards** node.

2 Click **Add text** from the **Actions** panel.

The **Entering User Text** dialog box appears.

Figure 10. Entering User Text



3 Choose to:

- Change the text **Style** (for example, Small)
- Type the **Text** to appear in the dashboard.
- Add a **Background Image**. For more information, see Adding a Background Image to a portal or its text component on page 19.

4 Click **OK**.

Delete This Page

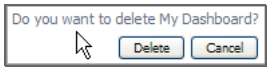
To delete the dashboard:

1 If the dashboard is not already opened, select the dashboard from the **My Dashboards** node.

2 Click **Delete this page** from the **Actions** panel.

The dialog box appears.

Figure 11. Deleting a dashboard



3 Click **Delete** or **Cancel**.

If you click **Delete**, the dashboard is deleted immediately. There is no additional confirmation dialog.

Adding a Background Image to a portal or its text component

In conjunction with the fixed position layout, you can set on the portal a background image, such as a geographical map or a layout diagram of your server rack, and then position a state indicator view at a location on top of the image to represent the monitored state at that location.

By dragging a text component you can upload a background image, such as a map.

To add a background image to a portal or a text area component:

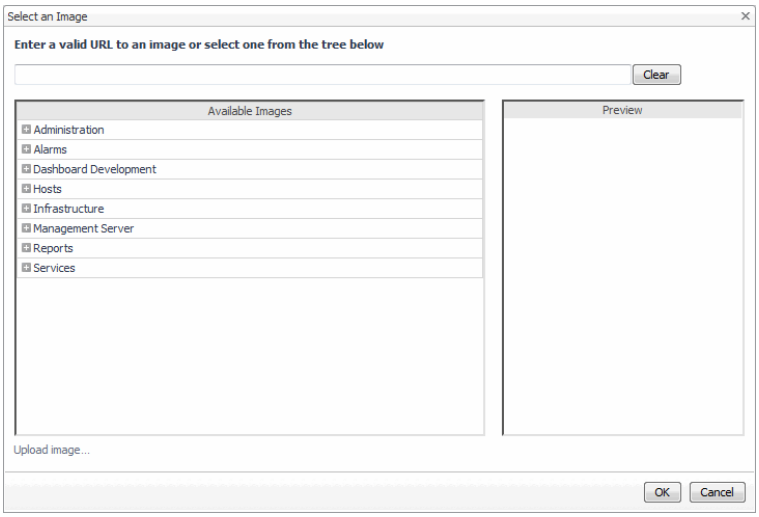
1 Click the **Background Image** icon to upload a background image.

- For a portal, the choice is available on the General tab by choosing **Properties > Edit basic properties**. The **Edit View Properties** dialog box appears.
- For the text component, the choice is available on the **General** tab by choosing **Drag into portal to add text**. The **Entering User Text** dialog box appears.

In either case the icon for setting a background is available to allow you to choose among the existing registered images or to specify a URL that points to the image.

The **Select an Image** dialog box appears.

Figure 12. Select an Image



2 Choose an option for selecting a background image:

- In the tree under **Dashboard Development > Support > Maps**, choose one of the mapped, themed images that are included with Foglight.
- Upload your own image by clicking **Upload Image**.

3 Preview the image and click **OK**.

You can use the image to add monitoring views based on absolute location by dragging and selecting views onto your image. For more information on adding views, see Add View on page 17.

Using the Metrics Analyzer

For any visualization that you create dynamically, there are built-in, default drill-downs available so you can go to another view. You can use the metrics analyzer to view the data in different ways by experimenting with metrics available in that same parent.

To use the metric analyzer on a chart:

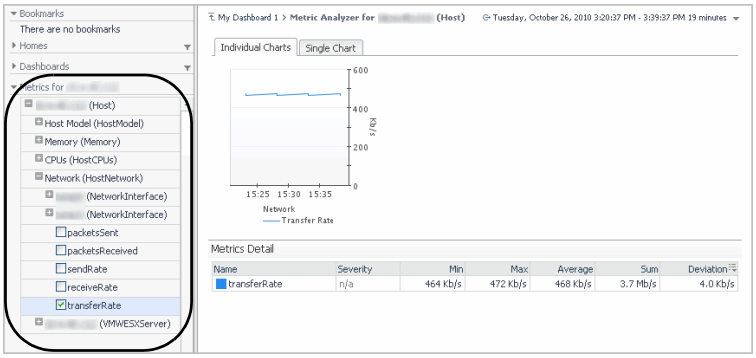
1 For any chart you have created, click the data in the chart.

IMPORTANT: The metric analyzer can be used with any metric, not just with charts.

Figure 13. Metric charts

The Metric Analyzer view appears with other metrics for the selected chart.

Figure 14. Metric Analyzer



2 Select other metrics so you can view them in that same parent. For example, if you select **receiveRate**, the corresponding chart also appears in the Metric Analyzer view.

3 Select the **Individual Charts** tab to display each metric individually or click the **Single Chart** tab to view all selected metrics in a single chart.

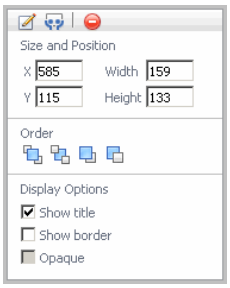
NOTE: From any view you have created on your dashboard, you can click the **Options** icon ( ), (also known as the **Edit** or **Customizer** icon depending on its usage). A popup appears that displays a list of other options available.

NOTE: You can also customize your view to add a configurable drill-down so you can go to another existing view or a drill-down you have created. For example to configure a user action so that a popup appears when a metric is selected, select **Edit Properties**. In the Config Portlet dialog box, click the **Actions** tab and choose the Popup option. For more information on configuring actions, see Actions tab on page 26.

Edit Page Layout

If you chose the Fixed position Layout option when creating a custom dashboard, you can use the settings in the **Edit Page Layout** dialog box to select a fixed position. A fixed position means that your view appears exactly where you want it.

Figure 15. Edit Page Layout



Using the Edit Page Layout dialog box, you can put the view exactly where you want it and tweak the layout so you can maximize the view with as much information as you prefer, making the most out of the real estate you have available on the screen. You can control the layout of a view or table on your dashboard by choosing a visualization style, how much space the view takes up and exactly where you want it to appear such as:

- **Size and Position**—resizes the co-ordinates of the view such as apply a width and height.
- **Order**—controls the stacking order such as bring the image to front, to back, forward, or backward.
- **Display options**—specify if you want to show title, show border, or make the display opaque.

Editing views

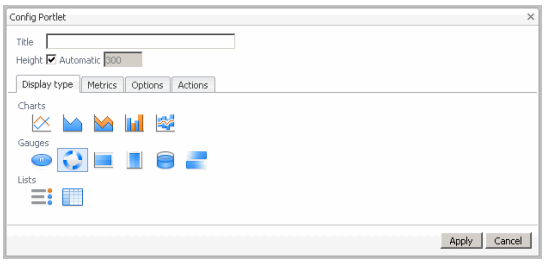
After you have set the initial settings based on “Creating a Custom Dashboard” in the Foglight® *User Help* and selected the layout position as fixed position, you can configure the portlet for additional views such as the display type of views, select metric labels, configure view settings, and choose the action when a user selects or dwells over a metric.

To edit the portlet view:

1. Select the custom dashboard containing the view you want to edit.
2. In the view that you want to edit, in the top-right corner, click Options , and choose **Edit Properties** from the menu that appears.

   The **Config Portlet** dialog box appears.

Figure 16. Config Portlet



3 On the Config Portlet dialog box, you can make changes on options such as:

- **Title**—rename the view.
- **Height**—set the height for all views or this view by selecting the check box and specifying a value.
- **Display type—**select the visual display. For more information, see Display type tab on page 22.
- **Metrics—**choose the metric labels for the view. For more information, see Metrics tab on page 22.
- **Options—**configure settings according to the display type. For more information, see Options tab on page 23.
- **Actions—**Set what happens when a user interacts with the view. For more information, see Actions tab on page 26.

IMPORTANT: The options that are available on the Config Portlet view depend on the type of view you are editing. The options listed in step 3 are only an example.

4 Click **Apply** when you are finished editing your dashboard.

Display type tab

The Display type tab contains options for selecting the visual layout according to a chart type, gauge type, or list type. For more information on the display types available and their settings, see Options tab on page 23.

Metrics tab

The Metrics tab displays the metrics you have selected to appear as a Metric label. If the chart contains metrics from more than one parent, the parent label also appears. For more information, see “Grouping Metrics with Many Parent Hosts” in the *Foglight User Help*.

Figure 17. Metrics tab

The settings that you can change are described in the following table.

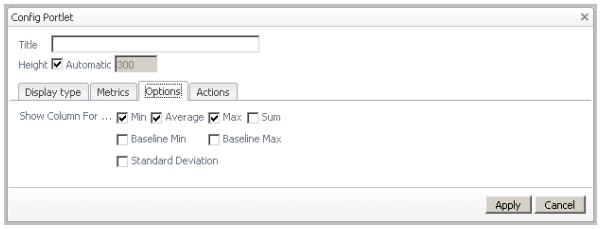
Table 1. Metric settings

Setting	Description	Options/Limits
Metric label	Allows you to choose another label for the metric.	Only available when editing a single chart.
Parent label	Allows you to choose another label for the parent metric. This label appears in the title of a chart that contains metrics from a single parent. The parent label appears in the legend when the chart contains metrics from more than one parent. (For example, see “Grouping Metrics with Many Parent Hosts” in the *Foglight User Help*).	Only available when editing a single chart.

Options tab

The options that appear here depend on the view type you are working with.

Figure 18. Options tab



The following tables lists the options available depending on the view type:

- Charts (see page 23)
- Gauge (see page 25)
- List (see page 26)

Charts

Table 2. Chart types

Type	Setting
• **High/Low** : Emphasizes the highest and lowest values.	**Show Average Line**: Indicates whether to show average values in the chart.

Table 2. Chart types

Type	Setting
• **Plot** : Draws each series as a line graph. • **Area** : Draws each series as connected points of data, filled below the points) • **Stacking area** : Draws each series as a stacked cluster) • **Bar** : Draws each series as a bar in a cluster)	**Line width**: Specifies the thickness of lines in all charts or this chart. Values available for selection: • **Automatic** • **Small** • **Medium** • **Large** **NOTE:** These options are available for Plot charts only. Other chart types do not include these options.
	Metric value: Controls the type of value that is displayed for one metric or all metrics in a chart. Values available for selection: • **None**. • **Average**: The chart displays the average value for the metric per interval. • **Minimum**: The chart displays the actual lowest value for the metric per interval. • **Maximum**: The chart displays the actual highest value for the metric per interval. • **Minimum/Maximum**: The chart displays the maximum and minimum value for the metric per interval. • **Sum**.
	Show one chart per metric: Indicate if you only want to display one chart per metric.
	Only show axis of selected metric (different charts will line up): Select this check box if only one axis is used in the chart, regardless of how many metrics are displayed. (For examples, see “Grouping Metrics with Many Parent Hosts” in the *Foglight User Help*.) Clear the check box to display the label for all metrics in the legend.
	Show thresholds for selected metric: If a metric has a threshold, it is displayed. You can choose to show thresholds only for the selected metric.

Table 2. Chart types

Type	Setting
	Show data at both start and end of intervals **NOTE:** This option is only available with Plot and Area charts.
	Honor Bounds: Sets the chart axis so that it does not go out of chart boundaries. This check box is selected by default.
	Show Overall: Displays the overall value for the set time range as a dashed line. Values available for selection: • **Average**: Displays the overall average. • **Min**: Displays the overall minimum. • **Max**: Displays the overall maximum.
	Show Min/Max As: Displays the minimum and maximum per interval. Values available for selection: • **Envelope** (a filled-in area between the minimum and maximum) • **Marks** • **Lines**
	Show Baseline Min/Max As: Displays the baseline minimum and maximum. Values available for selection: • **Envelope** (a filled-in area between the minimum and maximum) • **Marks** • **Lines**
	Show Standard Deviation As: Similar to the Min/Max setting; lets you highlight a range per interval. Values available for selection: • **Envelope** (a filled-in area between the minimum and maximum) • **Marks** • **Lines**
	Standard deviation multiplier: Determine the high and low values by setting the deviation from the average. The default value is `1`, but the unit of deviation depends on the metric.

Gauge

Table 3. Gauge types

Type	Setting
• **Metric Indicator**	**Indicator Size**: Useful for highlighting a fluctuating value in your real-time application in a visually meaningful way. Values available for selection: • **Normal** • **Wider**
• **Spinner** : Represents data using a rotating wheel.	No settings.
• **Horizontal Bar** : Horizontal bar, Vertical bar, and Cylinder give you flexibility to present data in terms of a graduated scale.	**Bar Thickness**: Sets the thickness of the bar in the chart. Values available for selection: • **Thinner** • **Normal** • **Thicker**

Table 3. Gauge types

Type	Setting
• **Vertical Bar**	No settings.
• **Cylinder**	No settings.
• **Flow Gauge** : Communication between host server components are represented by animated data flow that indicate the direction of the communication.	**Show Sparkline**: Indicates whether to show a sparkline.
	Sparkline displays: Indicates the type of values displayed by the sparkline. Values available for selection: • **Average** • **Minimum** • **Maximum** • **Extreme**
	Flow Direction: Indicates the horizontal direction of the flow: left or right. Values available for selection: • **Left** • **Right**
	Prominence: Specifies the thickness of the sparkline. Values available for selection: • **Less** • **Normal** • **More**

List

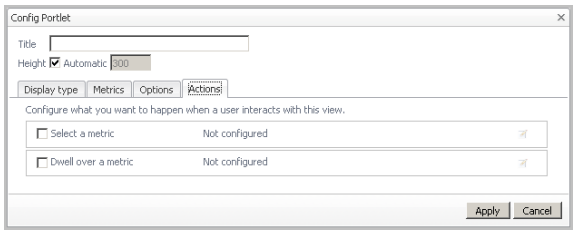
Table 4. List icons

Icon	Setting
• **Metric List** : Shows a list of metrics and their values. For more information, see Metrics tab on page 22.	**Show Sparkline**: Indicates whether to show a sparkline.
• **Table** : Shows a list of metrics and their values. For more information, see Options tab on page 23.	**Show Column For**: Indicates which columns to display in the table. Values available for selection: • **Min** (selected by default) • **Average** (selected by default) • **Max** (selected by default) • **Sum** • **Baseline Min** • **Baseline Max** • **Standard Deviation**

Actions tab

For charts and metrics that you have added to the dashboard you are creating, you can set the flow action to control what happens when a user selects a metric or dwells over it. For example, you can use the **Actions** tab to configure a popup, dwell, or a drill-down behavior, either to views that already exist in your system or other dashboards your have created.

Figure 19. Actions tab



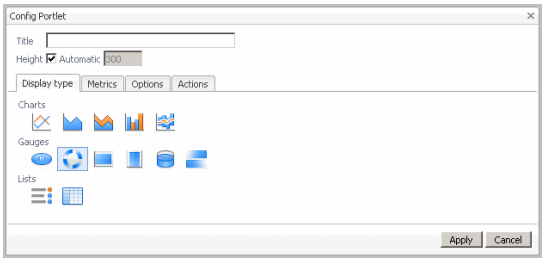
To configure an action on view selection:

1 Select the custom dashboard containing the view you want to edit.

2 In the view that you want to edit, in the top-right corner, click Options, and choose **Edit Properties** from the menu that appears.

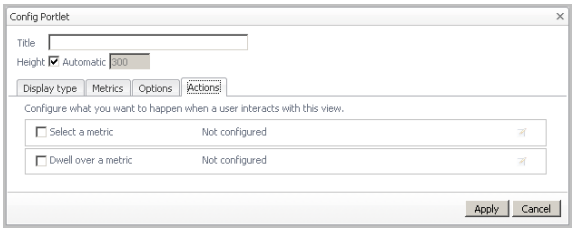
The **Config Portlet** dialog box appears.

Figure 20. Config Portlet



3 In the **Config Portlet** dialog box, open the **Actions** tab.

Figure 21. Actions tab



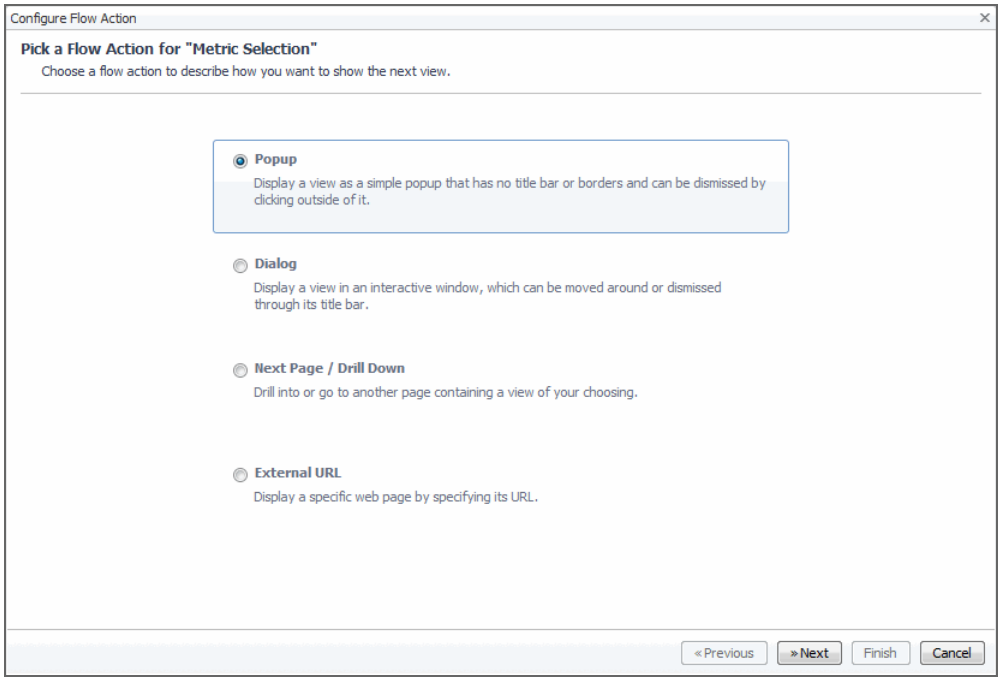
4 On the **Actions** tab, select the check box for the action(s) you want to set.

The Edit icon ( ) is enabled.

5 Click .

The **Pick a Flow Action for "Metric Selection"** wizard appears.

Figure 22. Pick a Flow Action for “Metric Selection” wizard



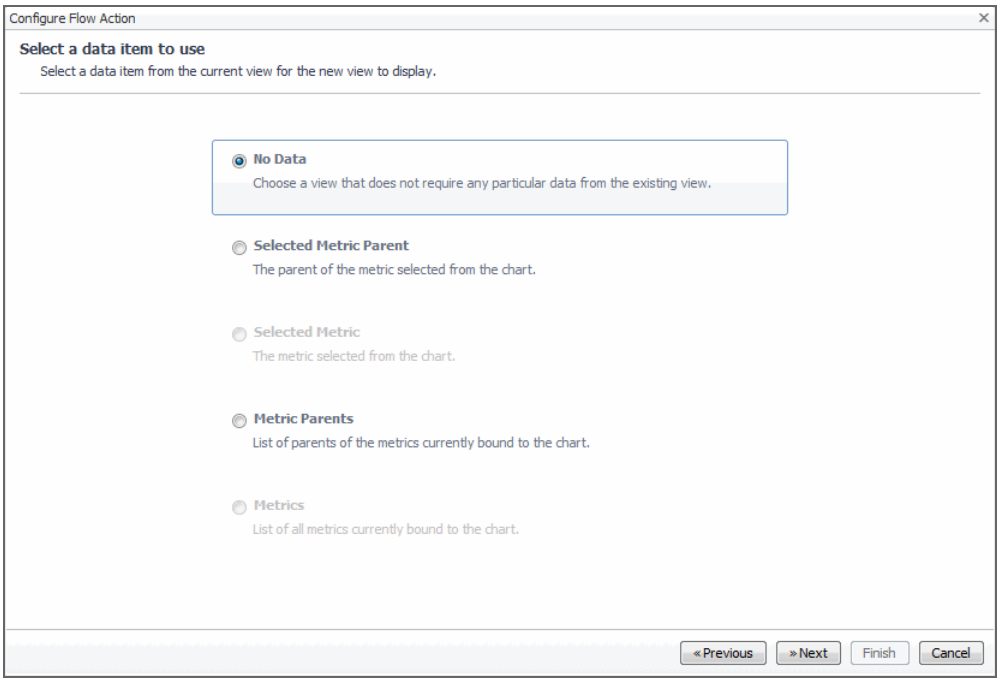
6 Choose a flow action to show how you want the next view to appear. For example, as a popup, dialog, drill-down or displayed as a specific URL.

7 Click **Next**.

The dialogs that appear next depend on your selection in Step 5.

- If **Popup**, **Dialog**, or **Next Page / Drill Down** is selected in Step 5, the **Select a data item to use** page appears in the wizard.

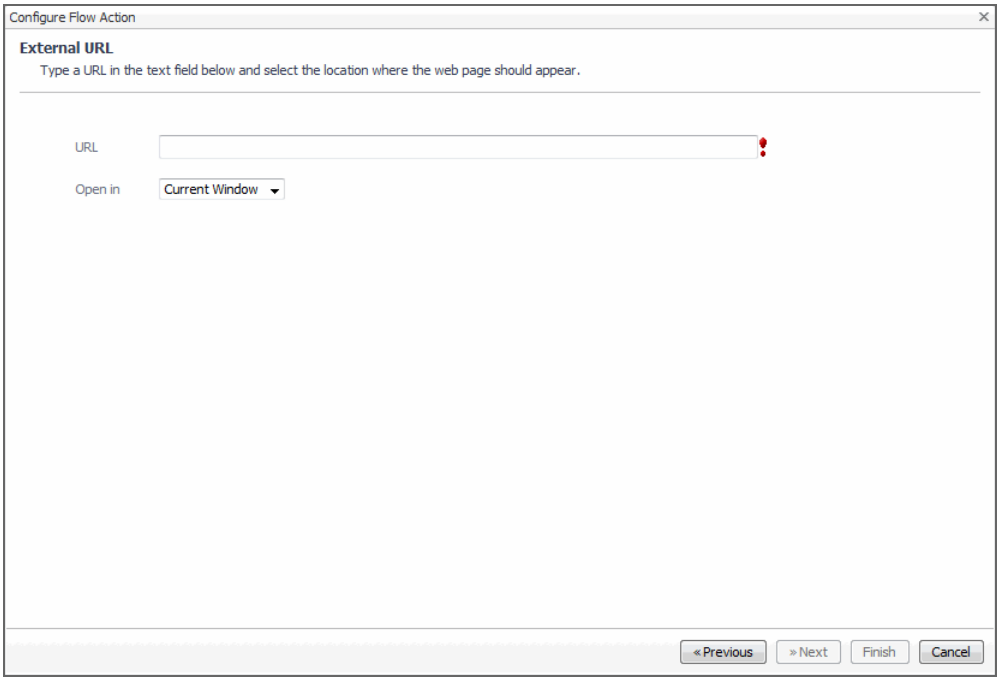
Figure 23. Select a data item to use page



Select the data item that you want to display in the new view.

- If **External URL** is selected in Step 5, the **External URL** page appears in the wizard.

Figure 24. External URL page



Type a URL and choose where you want the view to be displayed (for example, current window or a new window).

8 Click **Next** to continue with configuring the display of the selected views.

Sharing the view

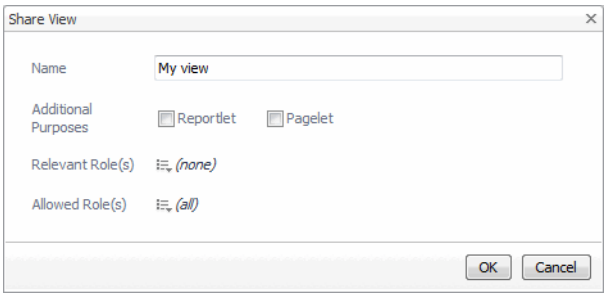
To share the view with other users on the same host:

1 Select the custom dashboard containing the view you want to share.

2 In the view that you want to edit, in the top-right corner, click Options, and choose Share from the menu that appears.

The **Share View** dialog box appears.

Figure 25. Share View



3 Specify the following:

- **Name**: The name of the view.
- **Additional Purposes**: Use these options if you want this dashboard to appear in another view such as:

Pagelet: Available for selection in pages or dashboards.

Reportlet—Available for selection in reports.

4 Click **OK** to save changes.

Removing the view

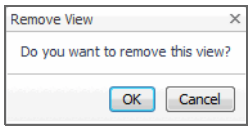
To remove the view:

1 Select the custom dashboard containing the view you want to remove.

2 In the view that you want to edit, in the top-right corner, click Options , and choose Remove from the menu that appears.

The **Remove View** dialog box appears.

Figure 26. Remove View



3 In the **Remove View** dialog box, click **OK** to confirm that you want to remove the view.

The view is deleted from the dashboard.

3

Tutorial 2: Creating a Dashboard

The term dashboard in Foglight® denotes a container that includes a pre-set collection of visual data, but does not necessarily require any user interaction to complete the picture. Technically, a dashboard cannot have any required context inputs, although it can have additional context entries. A dashboard presents an overall view of the health of a monitored system through the use of colored buttons, tabular data, various icons representing system alerts, and other graphical components. It supports a drilldown mechanism to show Web pages containing more detailed information about a selected item.

This tutorial shows you how to create a dashboard that shows a table of all the hosts in the monitored system.

User actions that cause the view to update will be covered in a subsequent tutorial.

- Constructing the Tutorial Module
- Configuring a Query
- Configuring a Row-Oriented Table
- Building the Dashboard
- Summary

Before you start

The first thing you have to decide is whether you want to take the top-down or the bottom-up approach. In the top-down approach, you create the dashboard first, then you populate it with components. In the bottom-up approach, you focus on the information you want to display, which forces you to think about which query or queries you want to build and how you want to present the results, such as in a table, or a chart, or some other component. The dashboard is a container that holds the components, so you define it after knowing what it will contain.

Since this is a tutorial, we will do a bit of both approaches. Our stated goal is to display a hosts table in the dashboard. That is our design for the moment. It is not really a design, just a bit of an outline, but it will serve to guide us as we build the view.

With that much of a goal in mind, we start the bottom-up approach with a query. Next, we use the query to populate the rows of a table and then add the table to an empty dashboard.

Anatomy of a typical dashboard

- A dashboard usually consists of a container, which in turn contains views.
- Views are assembled from View Components whose properties are listed in the *Web Component Reference*, which is accessible from the **Help** tab on the action panel.
- View components are often required to display specific data, which may be set at design time or may be based on dynamically-generated bindings. It is possible to configure a dashboard to respond to user interaction, which demands, in addition to input components and query-based data retrieval, a context mechanism for passing run-time information back to the dashboard so that it can update itself, or for passing this information on to child pages.
- In the Web Component Framework, the query mechanism allows you to retrieve data from a data source, for example, Foglight data from the Monitoring data source.
- Queries can be parameterized, allowing them to be modified at run time.
- A flow mechanism permits pages to be updated or linked to other pages.

> **IMPORTANT:** When returning from a child page, use the breadcrumb trail, not the browser's back button.

- A context mechanism allows values, which may be objects, to be passed to dependent pages. Thus, dynamically-retrieved data on a parent page can be passed to a dependent page.

The starting place for working with View Components is to choose, from the left panel, in the Dashboards group, **Configuration** > **Definitions**. Make this selection to follow along with the tutorial. (Your user module is labeled **My Definitions**.)

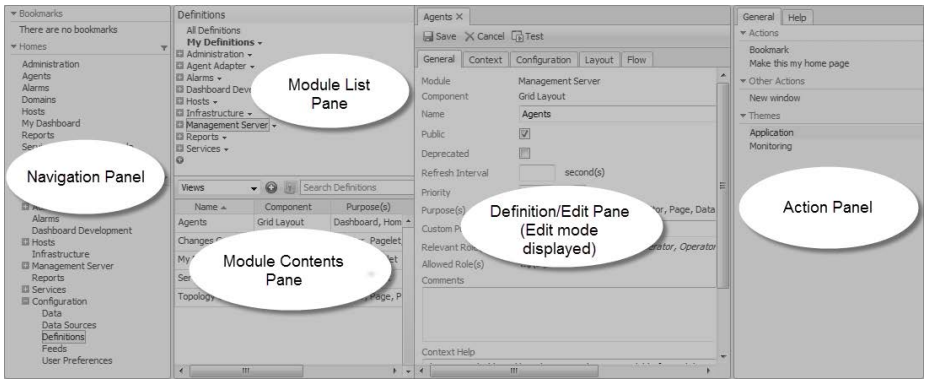
Browser interface walkthrough

The **Configuration** > **Definitions** page contains all the functionality needed to create new customized views, as well as allowing anyone with proper user permissions to edit existing modules. The Module List pane at the upper left side of the window lists all the modules for which definitions exist.

In this tutorial, we start off by configuring a user query in your user module **My Definitions**. First, select this module, and then focus your attention on the Module Contents pane. At the top of the pane on the left there is an Add button. You will use this button to add your new query. Above the Add button there is a drop-down list. The first two items are Views and Queries. We will be working with these items in this part of the tutorial.

We are going to start by defining a query, so ensure that the Queries choice is selected. When it is, the *Module Definition* pane on the right side of the window will be used to configure the query.

Figure 1. Browser interface elements



Constructing the Tutorial Module

There are two ways to work through this tutorial. The recommended way is to build all the queries and views yourself. That will give you the experience you need to become comfortable with the Web Component development environment. If you are pressed for time, you can read the tutorial and examine the pre-built components.

> **IMPORTANT:** If you are a novice at dashboard development, it is a good idea to create the queries and views yourself by following the instructions in the tutorials. You can create a new user and log on as that user. For example, you can configure this new user as tutorial and make this user a Dashboard Designer. Anything you create as this user will be available under My Definitions in the (Module List) Definitions pane. Any dashboards you create will be available under My Definitions in the navigation pane once you have saved them.The pre-built tutorial files can be made available in the navigation pane under Dashboards without constructing them yourself, but first you or the system administrator needs to assign the Dashboard Developer role to the tutorial modules in the Dashboard Support > Tutorials.

To make available the query and dashboard described in this tutorial:

1 On the navigation panel, choose **Dashboards** > **Configuration** > **Definitions**.

The Definitions area appears.

2 Navigate to **Dashboard Support** > **Tutorials**.

3 Click the triangle icon at the right of **Tutorial 1**.

4 Choose **Edit** from the menu that appears.

5 In the **Edit Module** dialog box that appears, on the right of **Relevant Role(s)**, click .

6 In the dwell that appears, select the **Dashboard Designer** check box, and then click **Apply**.

7 Click **OK** to close the **Edit Module** dialog box.

8 **Optional**. Refresh the browser interface.

On the navigation panel, the Dashboard Development node now contains the S**upport > Tutorials > Tutorial 1** node.

Configuring a Query

Web Component Reference queries are similar to SQL queries and perform a similar function. They are used to select a subset of the information in a data source. Foglight® agents collect a vast amount of data, so Web Component Reference queries are used to access specific items. The *Web Component Reference* building blocks are used to present this information in a readily understandable way.

Recall that these queries return objects. In most cases you want to use only a portion of the data returned by the query.

The query we want to define extracts from the data source information about all the hosts in our monitored system. To accomplish this task, it is helpful to understand something about Data Source Types, Data Source IDs, Object Types, and paths in the data object graph.

NOTE: There are other parameters that may be used with a query, such as Aggregations, filtering the top N results, Order By, and the familiar where clause. These parameters are not needed for the simple query we are constructing as our first example, so we will not discuss them here. The Web Component Guide contains more information about these topics.

To build a query:

1 In the Module Contents pane, click the drop-down list, select **Queries**, and click Add .

The **Add Query** dialog box appears.

2 In the **Add Query** dialog box, on the **Create a Blank Query** tab, in the **Name** column, click **Host** (with **Monitoring** in the **Namespace** column).

The list of data object types on this tab contains the names of all the object types known to this running instance of Foglight. You can search the by typing `Host` in the search box. list box responds to keystrokes, so you can type `Host` to quickly locate the **Host** type.

Click **OK**.

The Definitions pane displays the query editor, which contains the information needed to construct a query. You do not have to fill in every box, but the required boxes are marked by an exclamation point icon on the right side of the field.

3 In the **Name** box, type `Active Hosts`.

You will use this name later on to refer to the query.

4 Leave the **ID** box set to `(auto)`.

IDs are used internally by the browser interface mechanism. There is no need to change this default value at this point.

5 Ensure that the **UI Query** and **Public** check boxes are clear.

- Selecting the **UI Query** check box causes the query to appear as a node in the **Data** area. This is only useful if you want users to be able to explore the data objects returned by this query.
- Labeling a query as public allows it to be used in other modules. Since the tutorial is not meant to extend Foglight's functionality, your queries and views do not need to be used in other modules.

6 Ensure that the **Deprecated** check box is clear.

Selecting this box indicates that the view is likely to be deleted or replaced by something else in the future. In addition, if selected, this property prevents the component from being used as a child component in new views.

7 Leave **Relevant Role(s)** and **Allowed Role(s)** unchanged.

This makes the query available to everyone without restriction. See the section on roles in the *<Emphasis>Web Component Guide* for more information about setting roles.

8 Observe that the **Object Type** is set to `foglight-5:Host`.

9 Click the **Root Reference** box and choose **Data Sources > Monitoring (Default Instance)** from the list that appears.

10 Choose the object group for the query. Objects are grouped at some level down from the root. In this example, the object we want is in `ModelInstances`.

Click the **Path** box and choose **ModelInstances** from the list that appears.

11 Select the **Ignore Reference Objects** check box. This limits the size of the returned host objects to those sub-objects and properties that are strictly contained by a host.

An example of a contained object is an agent installed on a host. However, this same agent contains a back reference to the host containing it, so if **Ignore Reference Objects** is cleared, back references such as these are shown as nodes underneath the agent node. There are cases where this is the desired behavior, but it is not so in this tutorial.

12 Leave everything else as is, except possibly **Comments** and **Context Help**.

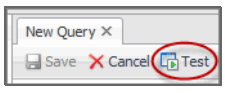
> **IMPORTANT:** The Comments and Context Help boxes are for describing the purpose of the query to developers and end users. You can supply descriptive text if you want, but they are not needed for the tutorial. However, any real application requires them. These help fields are discussed in the Foglight User Help.

At this point we have a valid query.

13 Test the query.

14 Click **Test**. This button appears at the top of the tab.

Figure 2. Test button



The **Query Results** dialog box appears, showing the name or names of the hosts. In the **Query Results** dialog box, the **Data Type** column indicates that all of the returned objects are of the `Host` type. The **Value** column is empty because there is no numerical value associated with these `Host` objects. Simple types have values, while `Host` objects do not.

> **IMPORTANT:** You can also see that the query returns a list of objects by expanding a Host object node in the Name column and observe the Value column for those properties that have numerical values associated with them. The Name column allows you to see all the properties of the object, which is useful if you want to determine their names for use later on.

15 In **Query Results** dialog box appears, click **Close**.

16 On the **New Query** tab, click **Save** to save the query. The tab name changes to **Active Hosts**.

Configuring a Row-Oriented Table

After completing the task in Configuring a Query, you have a query that returns a list of hosts. The next step is to use the query to populate the rows of a table.

To construct a Row-Oriented Table:

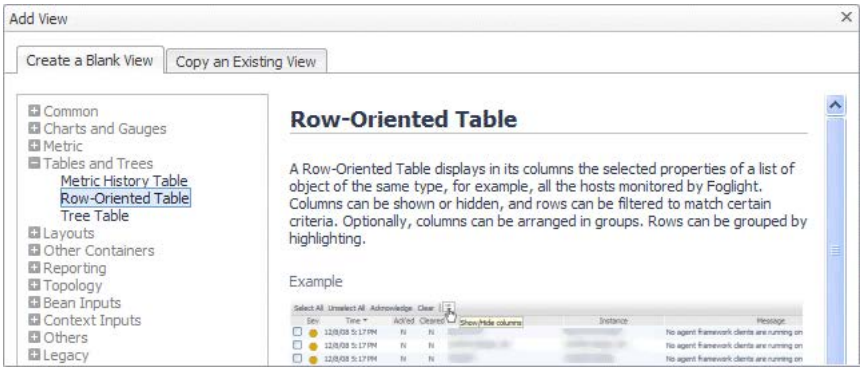
1 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

The **Add View** dialog box appears.

2 In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.

3 On the **Create a Blank View** tab, choose **Tables and Trees > Row-Oriented Table**.

Figure 3. Create a Blank View tab



4 Click **OK**.

The Definitions pane displays the view editor, which contains all the boxes that can be used to construct a view. The required fields are marked by an exclamation point icon at the right side of the field. In this case, only the **Name** field is required.

5 In the Definitions pane, on the **General** tab, in the **Name** box, type `Monitored Hosts`.

Later in this tutorial, you will use this name to refer to the Row-Oriented Table component.

6 Leave the **ID** box set to `(auto)`.

7 Ensure that the **Public** and **Deprecated** check boxes are clear.

8 Set the **Preferred Width** and **Preferred Height** values. You can edit these values later when you have seen the table displayed, so you can choose any values you want. For now, set *Preferred Width* to **300** and *Preferred Height* to **150**.

NOTE: Do not leave preferred sizes at 0. Set a non-zero number instead, as is done here.

9 Leave the **Refresh Interval** box blank.

To learn about the uses of refresh interval settings, see the *<Emphasis>Web Component Guide*.

10 Leave the **Priority** box set to **None**.

To learn about the uses of priority settings, see the *<Emphasis>Web Component Guide*.

11 Set the purpose of this view.

On the right of **Purpose**, click Edit ( ). In the dwell that appears, select **Pagelet**, and leave all the other check boxes cleared. Since the table will be placed on a dashboard, we need to ensure that view has the `Pagelet` purpose

Click **Apply** in the dwell to save this change and close the dwell.

12 Leave the **Custom Purpose(s)**, **Relevant Role(s)** and **Allowed Role(s)** values unchanged.

See the *<Emphasis>Web Component Guide* for more information about these settings.

The table is not fully configured yet, so if you try and save it now you will get an error that informs you that column values have not been set. The next procedure shows you how to choose columns for the table. The data in the columns' cells will be dynamically generated from the selected hosts.

13 Complete the **Comments** and **Context Help** boxes with any descriptive text, as required. In a production environment, comments should contain information for other developers and context help should contain information for end users. It becomes part of the context sensitive help for the component.

Configuring Table Columns

In the preceding steps you gave the table a name and you have set its size, but you have not populated the table. In other words, you need to define the columns for the table.

To configure table columns:

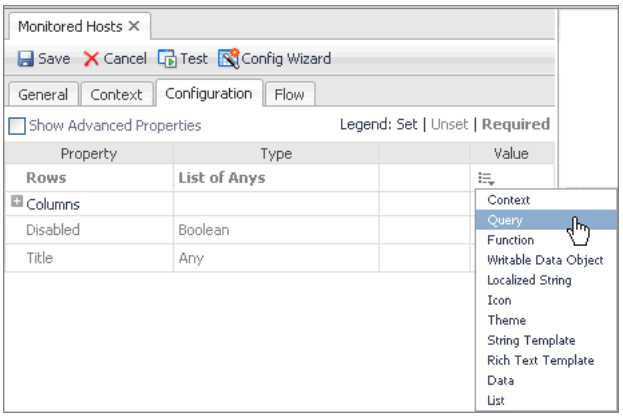
NOTE: This procedure continues from "Configuring a Row-Oriented Table" on page 36.

1 Open the **Configuration** tab.

The table designer page appears in the Definitions pane.

2 On the **Configuration** tab, ensure that the **Show Advanced Properties** check box is clear. The advanced properties are less likely to be used when constructing a basic version of the component which is what we are about to do in this procedure.

Figure 4. Configuration tab, Show Advanced Properties



3 In the table, locate the **Rows** row. In the Value column, click Edit , and choose **Query** from the menu that appears.

The **Edit - Rows** dialog box appears.

4 In the **Edit - Rows** dialog box, in the **Query** box, click drop-down list button, and in the navigation tree that appears, choose **My Queries > Active Hosts**. This is the query we created in Configuring a Query on page 33.

5 Without changing any other values, click **Set**.

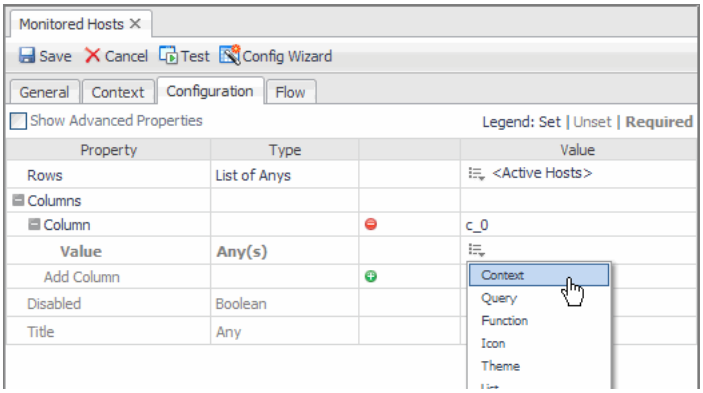
The table is now configured to retrieve its information from a query that returns a list of hosts. But we still have to define which property of a host will appear in a column. For that, we will have to choose the existing column and define its content.

6 In the **Property** column, expand the **Columns** node, and then the **Column** node that appears immediately below.

A **Value** node appears, along with an empty row with a Click to add button.

7 In the row containing the **Value** node, in the **Value** column, click Edit .

Figure 5. Value column, editing



8 From the menu that appears, choose *Context*.

The **Edit - Value** dialog box opens.

9 In the **Edit - Value** dialog box, in the **Input Key** box, click , and choose **currentRow (Host)** from the list that appears.

10 In the **Path** box, click , and choose **Name - String** from the list that appears.

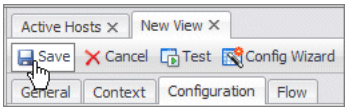
As each row is rendered at runtime, the context key *currentRow* contains a pointer to the current object in the list of hosts. Any property of a host object that can be rendered as a string can be selected as a column entry.

11 Click **Set**.

The **Edit - Value** dialog box closes.

12 Click **Save** to save your changes to the view.

Figure 6. Saving changes to the view



At this point you could continue to add columns, thus building a table that presents the data you want to see presented for each host. We leave the table for now and add more columns later.

Building the Dashboard

So far we have a query and a table to present some of the data that the query returns. Now we need to configure a dashboard. This gives us a container in which to place the table and other views that we might want to add later on.

To build the dashboard:

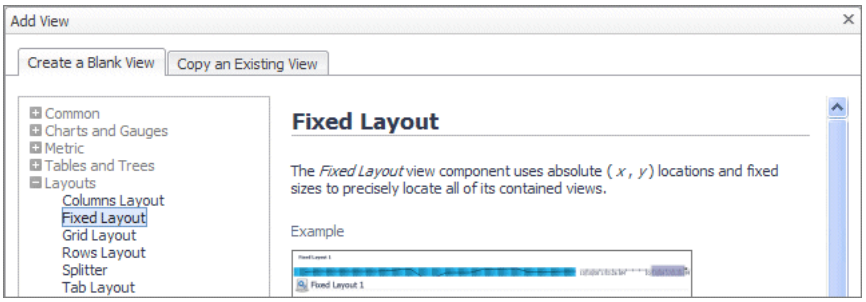
1 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

The **Add View** dialog box appears.

2 In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.

3 On the **Create a Blank View** tab, choose **Layouts > Fixed Layout**.

Figure 7. Choosing a fixed layout



Click **OK**.

The Definitions pane displays the view editor, which contains all the boxes that can be used to construct a view. The required fields are marked by an exclamation point icon at the right side of the field. In this case, only the **Name** field is required.

4 In the Definitions pane, on the **General** tab, in the **Name** box, type `All Hosts`.

5 Leave the **ID** box set to `(auto)`.

6 Ensure that the **Public** and **Deprecated** check boxes are clear.

7 Set the **Preferred Width** and **Preferred Height** values. You can edit these values later when you have seen the table displayed, so you can choose any values you want. For now, set *Preferred Width* to **300** and *Preferred Height* to **150**.

> **NOTE:** Do not leave preferred sizes at 0. Set a non-zero number instead, as is done here.

8 Leave the **Refresh Interval** box blank.

9 Leave the **Priority** box set to **None**.

10 Set the purpose of this view.

On the right of **Purpose**, click Edit ( ). In the dwell that appears, select **Dashboard**, and leave all the other check boxes cleared. Since this view is a dashboard container, its purpose needs to reflect that to be enable access to it later on.

Click **Apply** in the dwell to save this change and close the dwell.

11 Leave the **Custom Purpose(s)**, **Relevant Role(s)** and **Allowed Role(s)** values unchanged.

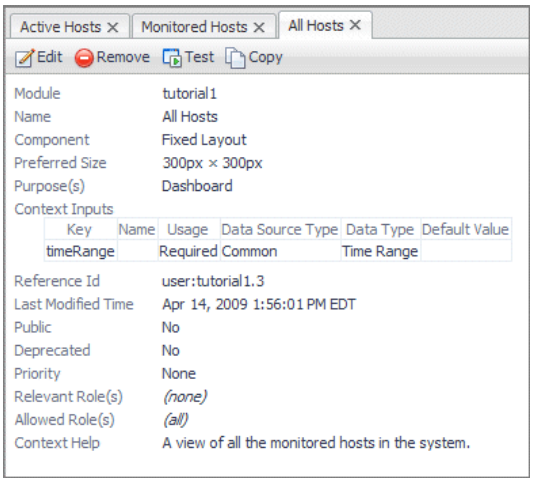
12 Complete the **Comments** and **Context Help** boxes with any descriptive text, as required.

13 Click **Save** to save your work so far.

The Definitions pane switches from edit mode to view mode and it presents the work you have done so far.

> **IMPORTANT:** Become familiar with the correspondence between the changes you made while in edit mode and the way those changes are presented in the view mode. Subsequent procedures in this tutorial show the contents of the view mode, which you use to make the required changes in edit mode.

Figure 8. Definitions pane in view mode



14 Click **Edit** to continue modifying the dashboard.

Adding the Table to the Dashboard

By performing the steps in the preceding section you have created a dashboard and you have set its size, but as yet you have not added any components to it. It is just an empty container. First we will give the dashboard a title.

To do that you use the *Configuration* tab.

To configure the dashboard:

NOTE: This procedure continues from Building the Dashboard on page 37.

1 Open the **Configuration** tab.

2 On the **Configuration** tab, locate the **Title** row, and in its **Value** column, click Edit .

3 In the menu that appears, choose **Set to Binding > String Template**.

The **Edit - Title** dialog box appears.

4 In the **Edit - Title** dialog box, in the **Value** box, type `All Monitored Hosts`, and click **Set**.

5 Open the **Layout** tab.

The layout editor appears in the Definitions pane.

6 On the **Layout** tab, click **Add**.

The **Add View** dialog box appears.

7 In the **Add View** dialog box, select **Select existing view**, and click **Next**.

The **Add View** dialog box refreshes.

8 In the **View** area, choose **My Views > Monitored Hosts**, and click **Next**.

The **Add View** dialog box refreshes.

9 Set the position and size of the table, indicate if you want scroll bars, and whether to show a title and a border. Check **Show Title** and **Show Border** and leave the other settings at their default values.

10 Click **Finish**. The **Add View** dialog box closes.

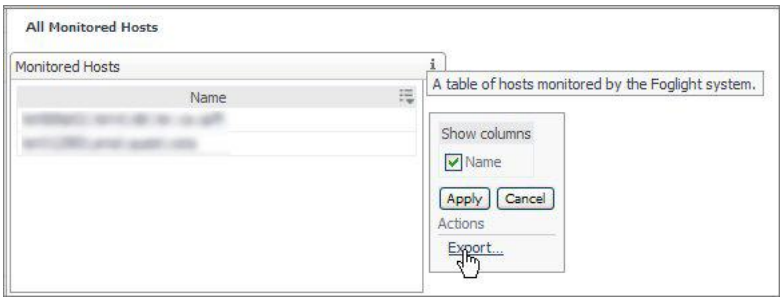
11 Click **Save** to save the changes to the view.

12 Observe the dashboard.

On the navigation panel, choose **My Definitions > All Hosts**.

The All Hosts dashboard appears in the display area. You see that **All Monitored Hosts** appears in the breadcrumb trail at the upper left of the view, and that the dashboard contains a table titled **Monitored Hosts**.

Figure 9. Accessing the newly built dashboard



Summary

In this tutorial you have constructed a query, used it to populate the rows of a table, and presented that table in a fixed-layout view component that we are calling a dashboard.

You have viewed the component by navigating to **All Hosts** in the *Dashboards* area under *My Definitions*.

When you build a dashboard page with a real purpose in mind, you want to present some useful information about each host, such as the state of applications running on it, and to see if there are any alarms. You also want to add your newly-constructed page to a place that already exists in the browser interface. Also, you may want to furnish a drilldown page that contains extra information about a particular host that you select from the table. To accomplish that, you need to understand how to use context inputs and flows. These topics are covered in the next tutorial.

Additional activities

Notice the small **i** in the table's top-right corner. Click it to see help for the table. Any text you placed in the **Context Help** box (see Step 13 on page 36) appears in a tooltip when the help icon is activated.

Notice the edit icon at the right-hand side of the table's column header. Clicking it launches a customizer. It isn't too useful right now because the table contains only one column, but if a table contains many columns this customizer gives the end user the option of temporarily hiding some of the columns. Also, the Export action allows the user to save a copy of the table in CSV, PDF, Excel and XML formats.

Because you gave the dashboard a title, it appeared at the top of the component. Experiment with giving the table another title.

To give the table another name:

1. Start editing the Monitored Hosts table, and open its **Configuration** tab.
2. In the row containing the **Title** property, in the **Value** column, click Edit , and from the menu that appears, choose **Set to Binding > String Template**.
3. In the **Edit - Title** dialog box that appears, in the **Value** box, type **Hosts Monitored by Foglight**.
4. Save your changes, both in the dialog box and in the view component, and observe the table title on the All Hosts dashboard.

To show or hide a title for the table:

1 Whether the table's title shows or not is dictated by a property in the All Hosts dashboard. Open it for editing and open the **Layout** tab.

2 On **Layout** tab, click anywhere in the **Monitored Hosts** area.

3 Click the **Properties** tab.

4 In the **Properties** dialog box that appears, clear the **Show Title** and **Show Border** check boxes.

> **IMPORTANT:** You can change the width of the table here, too.

5 Save the changes and view the result. Observe that there is no title and no border around the table.

6 Return to the **Properties** dialog box, select the **Show Title** and **Show Border** check boxes, save your changes and view the result. Observe the table title.

> **NOTE:** You may notice that the column containing host names has a heading whose value is Name, which is derived from the property chosen for this column in the context selection (<currentRow>/name). The header can be changed by editing the column's Header property. To do that, in Monitored Hosts table, select the Show Advanced Properties check box, and navigate to Columns > Column > Header.

Saving the Tutorial2 module

There is a command-line option in `fglcmd` that allows you to export and import modules. This procedure assumes that your Foglight® installation is in the default location on a Windows® machine. Make the appropriate changes for other platforms. It also assumes that your user name is `tutorial`. Note that all objects in the module are saved.

For complete information about the `fglcmd` utility, see the *Command-Line Reference Guide*.

To copy fglcmd to your machine:

- Log on to the server machine and type:
  `cd C:\Quest_Software\Foglight\tools`
  Copy *fglcmd* to your machine.

To export your module using fglcmd:

- In the directory where you placed *fglcmd*, issue the command:
  `fglcmd -usr tutorial -pwd <password> -cmd util:uiexport -m user:tutorial -f \C:\pub\Tutorial\Tutorial2.zip.`

To import a module you have exported:

- `fglcmd -srv <serverName> -usr tutorial -pwd <password> -cmd util:uiimport -f C:\pub\Tutorial\Tutorial2.zip`

> **IMPORTANT:** When you load a module it completely replaces the contents of the module. Any definitions you may have placed there are lost.

4

Tutorial 3: Adding a Drilldown Page

In Tutorial 2 you created a top level dashboard that contains a table of host names. Now, in Tutorial 3, it is time to show you how to add more columns to the table and how to create a dependent page that supplies information about any host selected from the table. This is often called a *drilldown page* because the information it presents depends on the object selected in the parent page.

User actions that cause the view to update in some way will be covered in a subsequent tutorial.

- Preliminary Setup
- Adding a Column of Aggregate Alarms to the Table
- Adding a Dependent Page
- Summary

Objectives of Tutorial 3

There are two main ways of presenting visual information about a group of objects. One is to list objects on a page and to organize it so that the information about each object appears with it on the same page. We chose a table layout in Tutorial 2 with this objective in mind. We can add other bits of information to the same row as the host and keep things together that way.

The other way of presenting information is as separate pages, one for each item in the original list. This way is appropriate when the visual component presenting the information is too large to fit on the parent page.

The objectives of Tutorial 3 are:

- Add columns to the table of hosts to show the aggregate alarm state. This places visual information about the host on the row adjacent to its name.
- Add a drilldown page that presents information about the chosen host. This permits you to show much more information about each individual host.

The primary objective of this tutorial is to demonstrate the addition of a drilldown page whose content depends on the particular choice made in the parent page, and in so doing, demonstrate the use of a context and a flow. The terms context and flow are described in detail in the <Emphasis>Web Component Guide.

Preliminary Setup

You are going to build on Tutorial 2 here, but it is a good idea to keep the work you did on Tutorial 2 intact so that you can reuse it if necessary. If you want to keep the tutorials separate, create a new user and copy Tutorial 3 to the new workspace. If you are creating views as you read this tutorial, it is a good idea to start fresh and copy Tutorial 2 to Tutorial 3.

To copy the Tutorial 2 module to Tutorial 3:

1. Create a new Foglight user (*tutorial3*, *tut0rial*), as you did for Tutorial 2.
2. Log in to Foglight as *tutorial3*.
3. On the navigation panel, choose **Configuration > Definitions**.
4. In the Module List pane, choose **Other User Definitions** > **tutorial2**.

5 In the Module Contents pane, click the drop-down list, and select **Views**.

6 In the list of views, select **All Hosts**.

The Definitions pane shows the configuration of the All Hosts view.

7 Click **Copy** and from the menu that appears, choose **Deep**.

8 In the **Copy** dialog box that appears, in the **Select a target module** area, choose **My Definitions** and click **OK**.

Since you are logged in as user *tutorial3*, you now have a separate copy of the module you created in Tutorial 2. You can use this copy to continue your work in Tutorial 3 without affecting the work you did for Tutorial 2.

To add a context entry to All Hosts:

1 In Tutorial 3, open the All Hosts view for editing.

2 Open the **Context** tab.

3 In the **Inputs** table, click Add Input .

4 In the newly added row, in the **Key** column, type `host`.

5 Leave the **Name** column unchanged.

You can set a name to be more descriptive than the key. This can be useful when configuring reports that require context inputs to be set. Using a descriptive name may help the person understand the purpose of the context key.

6 Click the **Usage** column and choose **Internal** from the list that appears.

Later in this tutorial, you will set a flow to Host Monitor in Monitored Hosts, and pass a key named `host` as the context mapping for the selected row that triggers the flow action. Setting the same context as `Internal` here tells the layout that one of its children supplies the value. This step is not strictly required for the current tutorial, but it is a pattern that should be followed as a matter of course because other situations require it.

It often happens that you design a dashboard that should update itself based on a choice the user makes in a contained view. For instance, you might define an Update flow on a row of a table in the dashboard. The context mechanism sends the key up to the dashboard, but without a context declaration at that level, the dashboard can not send the context back to the table it contains. Thus, the table does not receive any context information from its parent. In this case, the table uses its default value of the context from a time before the current row was selected. The new context information is lost, and the view updates itself based on the old context. This situation is avoided by defining an *Internal* context at the dashboard level. This tells the dashboard to retrieve the value for this key from one of its child components. Upon an update, the dashboard receives the key and sends it on to all its children. Those that define the same context key can use it.

7 Leave the **List** column set to **False**.

8 In the **Data Type** column, type `Monitoring:Host`.

9 Set a fallback value. Choose **null** and ignore the warning message because the context value will be used in a child view later on in this tutorial.

10 Click **Save** on the dialog box and on the tab on All Hosts' toolbar to save your work so far.

Adding a Column of Aggregate Alarms to the Table

There are two ways of adding columns. You can work with the `Columns` property as you did in when first creating the table in Tutorial 2, or you can use Config Wizard.

To add a column of aggregate alarms to the table of hosts using the Config Wizard:

1 On the navigation panel, choose **Dashboards** > **Configuration** > **Definitions**.

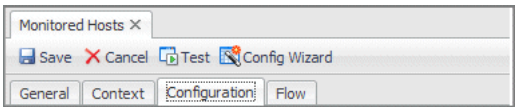
2 In the Module List pane, choose **My Definitions** from the list of modules.

1 In the Module Contents pane, click the drop-down list, select **Views**.

The group of views defined in this module appears.

2 Choose **Monitored Hosts** and open the view for editing.

Figure 1. The *Definitions* view pane fills with the parameters that currently define the table



3 Click **Config Wizard**.

The first page of the **Config Wizard** appears in the edit pane, allowing you to choose one or more properties that are available for selection.

4 In the **Config Wizard**, select the **Aggregate State Severity** node.

By choosing **aggregateStateSeverity** (an integer) we are asking for a value that represents the highest level alarm that is raised for this host. Even though this property contains an integer value, it is replaced by an icon by the default renderer associated with this data type. Thus, the icon shows up instead of the numerical value.

5 Click **Next**.

The **Config Wizard** refreshes.

6 Accept the default settings by clicking **Finish**.

7 **Optional**. Check the result by clicking the **Test** button.

To add a column of aggregate alarms to the table of hosts by configuring the Columns node:

1 Open the *Monitored Hosts* view for editing.

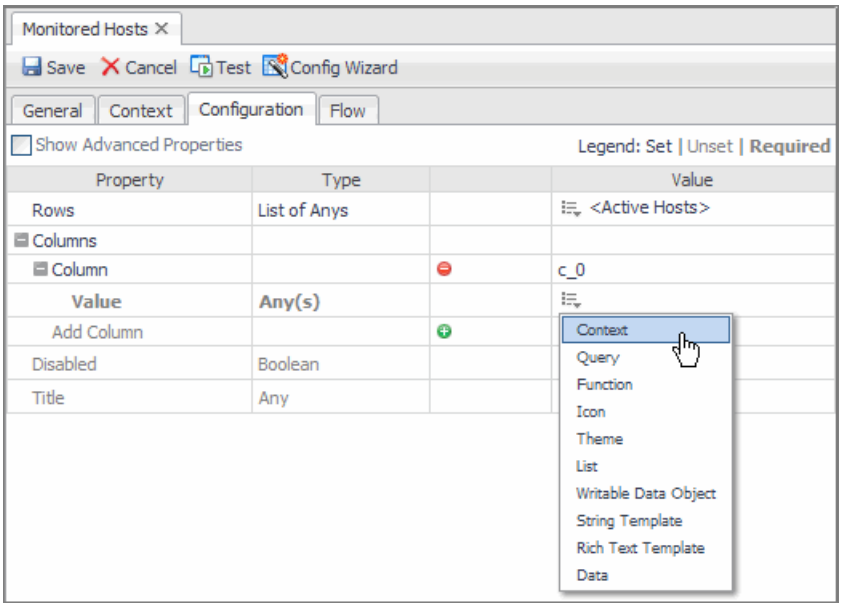
2 Open the **Configuration** tab.

This tab shows the configuration property editor for the Monitored Hosts view. Since none of the properties we want to modify are advanced properties, leave the **Show Advanced Properties** check box clear. This makes for easier viewing.

3 In the **Columns** row, expand that node and click .

4 In the **Value** row that appears, click Edit , and choose **Context** from the menu that appears.

Figure 2. Value menu entries



You are choosing **Context** because you want the value that appears to depend on the selected row in the table. Recall from Tutorial 2 that the row supplies a context that identifies a particular host: the one whose name appears in the selected row.

The **Edit - Value** dialog box appears.

5 In the **Edit - Value** dialog box, in the **Input Key** box, click , and choose **currentRow (Host)** from the list that appears.

6 In the **Path** box, click , and choose **Aggregate State Severity** from the list.

By choosing **Aggregate State Severity** (an integer) we are asking for a value that represents the highest level alarm that has been raised for this host. Even though this value is an integer, it is replaced by an icon by the default renderer associated with this data type. Thus, the icon shows up instead of the numerical value.

7 Click **Save** both in the dialog box and in the view editor to save the configuration changes.

When you view the All Hosts dashboard, you may notice that the area assigned to the table may not be wide enough to show all of its columns without clipping the right most column. If this happens, edit the All Hosts dashboard by opening the **Layout** tab, selecting the **Monitored Hosts** area, and dragging its right edge.

Adding a Dependent Page

This task introduces the notion of context inputs. In this example we use a *required* context input. Contexts are entities that have a data type, so a context must first be declared, much like a variable in a programming language, by giving it a key, which is the name by which it can be referenced, and a type, which is chosen from a list of available types. You will see in this example how a value is assigned to a named context so that it can be passed to the drilldown page through a flow.

Adding a dependent page that relies on a context supplied by the parent page involves these steps:

- Create a dependent page.
- Define a Context Input.
- Define a flow from the parent page to the dependent page.

- Set a Flow Context Mapping on the table row that supplies the context to the dependent page.
- Insert the desired component on the dependent page and specify the same context in the component that needs it.

Adding a Context-Sensitive Dependent Page for the Table of Hosts

In Tutorial 2, we choose a fixed layout container because it has simple properties that are easy to adjust. When, in this tutorial, we need to widen the table it contains, it is a simple matter of dragging the table's right edge in the **Layout** tab.

In this example we use a Grid to hold the component we plan to embed. The Grid container permits the placement of designated, already-existing views in cells. A later tutorial illustrates more of the Grid's properties.

When adding a view, you can specify the row and column in which to place it. The row and column indices are zero-based, so the top-left cell is in column 0 of row 0. We place a single component, a host monitor page, in this cell.

The Host Monitor page is quite complex, but it has the advantage of being one that already exists. You can find it in the **Hosts** module.

Create a dependent page

To create a dependent page called 'Host Monitor':

1 On the navigation panel, choose **Dashboards** > **Configuration** > **Definitions**.

The Definitions area appears.

1 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

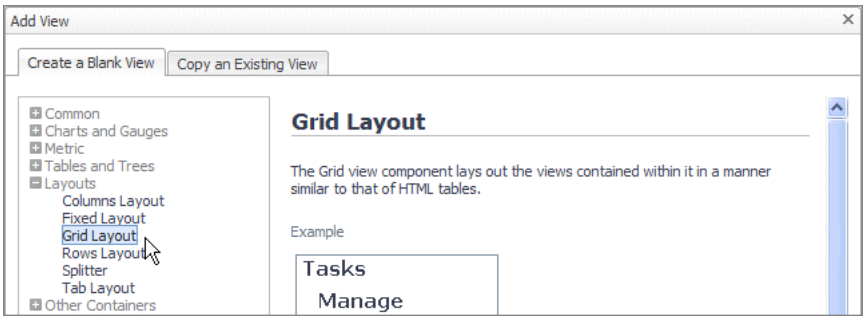
The **Add View** dialog box appears.

2 Click the **Add** button in the Module Definitions pane.

The **Add View** dialog box appears.

3 In the **Add View** dialog box, on the **Create a Blank View** tab, choose **Layouts > Grid Layout**.

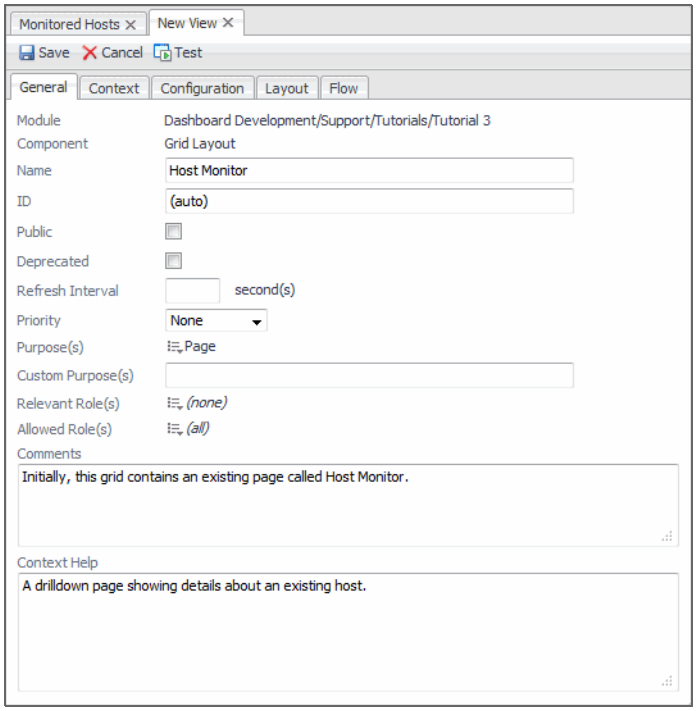
Figure 3. Add View



4 Click **OK**.

In the Definitions pane, on the **General** tab, populate the settings using the following values.

Figure 4. General tab



Define a context input

1 With the **Host Monitor** view open for editing, open the **Context** tab.

2 Under Inputs, click Add Input .

A new row appears in the table.

3 In the new row, in the **Key** column, type `host`.

4 Leave the **Name** column unchanged.

You can set a name to be more descriptive than the key. This can be useful when configuring reports that require context inputs to be set. Using a descriptive name may help the developer understand the purpose of the context key.

5 Ensure the **Usage** column is set to **Required**.

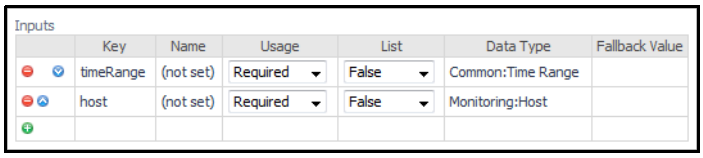
6 Ensure the **List** column is set to **False**.

7 In the **Data Type** column, type `Monitoring:Host`.

> **IMPORTANT:** By default, the timeRange context key is also passed to the component.

8 Leave the **Name** column empty. Ignore the warning when saving the component.

Figure 5. Context Inputs



9 Click **Save** in the top-left corner to save your work so far.

IMPORTANT: Ignore the warning when saving the component.

Define a flow

The following steps direct the rows of the table to a view called Host Monitor and pass the required context.

To configure the flow:

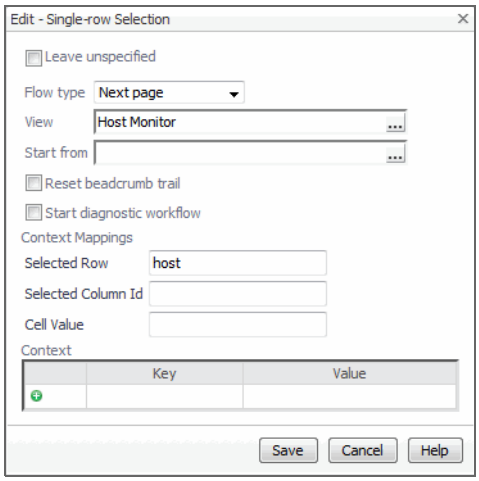
1. Open the **Monitored Hosts** view for editing.
2. Open the **Flow** tab.
3. In the table left-most column, click **Single-row Selection**.

   This establishes a flow action for every row of the table.

   The **Edit - Row Selection** dialog box opens.
4. In the **Edit - Row Selection** dialog box, clear the **Leave Unspecified** check box.
5. Set **Flow Type** to **Next Page**.

   This sets the flow type to launch a new page when a row of the table is selected.
6. Click the button on the right of the **View** box, choose **Host Monitor**.
7. Set **View** to **Host Monitor.**
8. Under **Context Mappings**, in the **Selected Row** box, type `host`.

   The target page has a required input of `host`. Here, you are specifying that the selected row pass this key. The Host Monitor view does not show up in the list of available choices unless its required input is satisfied.



9. In the **Edit - Single-row Selection** dialog box, click **Save**.

   The **Edit - Single-row Selection** dialog box closes and the **Single-row Selection** entry appears as a node, containing two sub-nodes: **On Missing Inputs** and **On View Not Found**. You can set a fallback view to provide an alternate if the drilldown page is not available.

   This component is your container for the Host Monitor view. You will add this and other components to your version of Host Monitor later.
10. Open the **Configuration** tab and select the **Show Advanced Properties** check box.
11. Under **Selection Configuration**, set **Allow Reselection** to `true`.

IMPORTANT: If you do not set this property to true, the selection action does not happen because it is already selected by default and the drill-down action does not occur.

12 Open the **Context** tab and under **Inputs**, observe a newly added `host` entry.

13 Click **Save** in the top-left corner to save the changes made to Monitored Hosts.

If you test the page at this point after choosing values for the bindings, you see that by clicking on a row of Monitored Hosts, you drill down to the Host Monitor view. The breadcrumb history trail at the top right side of the page refreshes accordingly, showing **All Monitored Hosts > Host Monitor**.

Populate the dependent page

To add a view to Host Monitor:

1 With the Host Monitor view open for editing, open its **Layout** tab.

2 Click **Add**.

3 In the **Add View** dialog box that appears, choose **Select existing view** and click **Next**.

4 Ensure the **Filter By: purpose** and **validity** check boxes are selected.

5 Choose **Hosts > Host > [chart] CPU Utilization**

6 Expand **Hosts > Host** and choose **[chart] CPU Utilization**, and click **Next**.

We chose an existing page that displays useful data about a host rather than attempting to build such a complex page from scratch. Any page whose purpose is `Page` and accepts context inputs of `timeRange` and `host` can be used. A later tutorial demonstrates how such a page can be constructed.

7 The next page of the **Add View** dialog box appears. It allows you to set properties for the view.

8 Accept the default setting for the **Width** and **Height** properties by selecting these check boxes, along with **Show Title** and **Show Border**.

9 Click **Finish** to close the **Add View** dialog box.

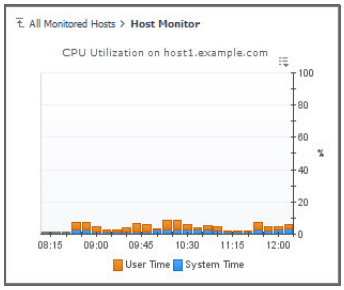
10 Click **Save** in the top-left corner to save the changes made to the Host Monitor view.

11 View the result by going to the **Dashboards** > **My Definitions > All Hosts**.

12 Click a row of the table to drill down to the **Host Monitor** view.

The figure shows a sample result. The CPU utilization for the selected host is displayed. Each different row selection in the table launches a view that presents data coming from the selected host. The overall construction of the view is always the same, but the data presented in the view comes from the chosen host.

Figure 6. Newly created page



Summary

In this tutorial you have been shown how to use a context input and a flow to present a drilldown page in which the data depends on the context that was passed to the dependent page.

Additional activities

After completing this tutorial, consider performing the following activities:

- Saving your work (see page 50)
- Adding a vertical scroll bar to the table (see page 50)
- Interchanging columns in the Table of Hosts (see page 50)
- Changing the name of the Hosts column (see page 51)

Saving your work

Just as you may have done in Tutorial 2, you can save the work you have done so far in Tutorial 3.

This procedure assumes that your Foglight® installation is in the default location on a Windows® machine. Make the appropriate changes for other platforms. It also assumes that your user name is *tutorial*. Note that all objects in the module are saved.

To export your module using fglcmd:

- In the directory where you placed *fglcmd*, issue the command:
  `fglcmd -usr tutorial -pwd <password> -cmd util:uiexport -m user:tutorial -f \C:\pub\Tutorial\Tutorial3.zip.`

Adding a vertical scroll bar to the table

If there are more than six or so hosts listed in the table, the lower ones will be cut off and you will not be able to see them. One way of allowing for this possibility is to add a vertical scroll bar. It is one of the configurable properties you can choose when adding the table to the layout component.

To add a vertical scroll bar to Monitored Hosts:

1. Open the **All Hosts** view for editing.
2. Open the **Layout** tab.
3. On the Layout tab, click the **Monitored Hosts** area, and then click **Properties**.
4. In the **Properties** dialog box that appears, experiment with the **Scrollbars** settings, especially **Auto** and **Vertical**.

Interchanging columns in the Table of Hosts

To move the alarm to the left side of the table:

1. Open the **Monitored Hosts** view for editing, and open the **Configuration** tab.
2. In the list of properties, expand the **Columns** node.
3. In the second **Column** row, click to move the column up in the list of columns.

   This has a result of shifting the second column to the left in the **Monitored Hosts** view.

To add one more column to the table between the alarm and the host name:

1. Open the **Monitored Hosts** view for editing, and open the **Configuration** tab.
2. In the list of properties, expand the **Columns** node.
3. Under the last column node click . Click the **Add Column** button.

   A new **Column** entry is added to the list. Its expanded view appears. In the **Value** column, click Edit.

4 In the list that appears, choose **Context**.

5 In the **Edit - Value** dialog box that appears, click the **Input Key** box, and choose **currentRow(Host)** from the list that appears.

6 Click the **Path** box, and choose **Alarm Total Count - Integer** from the list that appears.

7 In the **Edit - Value** dialog box, click **Save**.

8 In the newly added **Column** row, click to move the column up so that the **Alarm Total Count** column appears between the **Name** and **Aggregate State** columns.

9 Save your changes and view the result.

After adding a new column to the table while keeping your previous settings unchanged, and testing the result, you may notice that the table is too narrow to fit the name column in the space that has been allotted to it. In fact, since the column has been placed to the right of the host name, you may not even see it. This exposes a problem with a fixed-layout container. The table's allotted size can be inappropriate if columns are added or if the text in a cell needs extra space.

You can widen the table by editing the **All Hosts** view.:

1 On the **Layout** tab, click inside the **Monitored Hosts** area, and then click **Properties** on the **Layout** tab toolbar.

2 In the **Properties** dialog box that appears, change the **Width** to `400`.

3 Click **Save** in the **Properties** dialog box, and then click **Save** on the **All Hosts** view toolbar.

Changing the name of the Hosts column

To change the name of the hosts column:

1 Open the **Monitored Hosts** view for editing.

2 On the **Configuration** tab, click the **Show Advanced Properties** box.

3 In the expanded list of properties, expand the **Column** node whose **Value** property is set to **<currentRow>/name**, and in its **Header** row, in the **Value** column, click Edit.

4 In the list that appears, choose **Set to Value > String**, and in the Edit - Header dialog box that appears, type `Host Name` in the **String value** box.

5 Click **Set**.

6 Save the changes to the view.

TIP: Since you set **Show Title** on this component in **Host Monitor**, a default title (the component's name) appears above the component in the layout. You can set a different title by editing the **Title** property on the **Configuration** tab.

Tutorial 4: Adding Views

In Tutorial 2 you created a top level dashboard that contained a table of host names. In Tutorial 3, you created a dependent page that contained a pre-built view. In Tutorial 4, you add views that you create, and learn about four additional components: chart, button, drop-down list, and customizer.

- Designing the Page
- Preliminary Setup
- Defining the Alarms Query
- Configuring a Table of Alarms
- Configuring a Chart Component
- Configuring a Chart Component
- Configuring a Label with an Action
- Configuring the Drilldown Page
- Configuring a Drop-Down List
- Flowing Monitored Hosts to Host Details T3
- Adding a Customizer
- Adding Nested Views
- Summary

Objectives of Tutorial 4

The objectives of this tutorial are to place components on a drilldown page whose content depends on the particular choice made in the parent page, and also to introduce three new components for you to work with.

The tasks you will complete are:

- Add a drilldown Grid page that presents information about the chosen host.

  This first exercise is similar to the one from Tutorial 3, but in this case you populate the drilldown page with components that you configure rather than simply choosing a pre-built view.

- Add a Row-Oriented Table of alarms for the chosen host.

  The first column of the Row-Oriented Table will show the severity of the alarm and the second column will present an informative message about the alarm.

- Add a chart component that you build yourself that shows CPU utilization.

  The two most common views in any UI design are likely to be tables and charts. You already learned about tables. This exercise introduces you to using a chart as a visualization component.

  Charts that display information that depends on the context in which they are launched are an essential UI component.

- Add a Drop-Down List.

  Frequently, a user can look at a page that displays information about one host and then decides to look at another host. This exercise shows you how to place a chooser on the page that allows viewing a different host.

- Add a Label that provides an active link back to the parent page.

  That is what the history trail (breadcrumbs) at the top of the page is for, but since this is a tutorial, you see how to define a flow on a label that brings the user back to the top page. In a more elaborate design, the label can be used to open a page that is not in the breadcrumb trail.

- Add a Customizer to the page.

  This exercise introduces you to another linking mechanism, the Customizer.

- Add two nested views.

  A view that will never be used in any other container should be created as a nested view. This makes it clear to any other developer that the view is not meant to be used elsewhere.

NOTE: While investigating Time Range settings is not an objective of this tutorial, ensure that the default timeRange context key is set to its default value for all the components in this exercise: Key: timeRange, Usage: Required, Data Type: Time Range.The timeRange context is automatically set for every view. Do not remove it from the context list unless you are certain that the view and its children do not require it.

Designing the Page

The objectives for Tutorial 4 list the components we are going to use, so all that needs to be done to design the page is to specify its layout. The table below shows how the components will be arranged.

Host Details T3 dependent page for Tutorial 4: a Grid layout with two columns

Above the grid

A Customizer that places itself above and to the left of the View and Help buttons on the title bar.

Table 1. Cells in the grid

Row 0, Column 0, Column Span 2: Memory - Utilization History: A chart component.	
Row 1, Columns 1 and 0, Column Span 2: Alarms table for <host>: A Row-Oriented Table listing alarms for the chosen host. The component will span two columns of the Grid container, similar to this row.	
Row 2, Column 0: Return to Hosts Page T3: A Label component (with a flow action that returns to the parent page).	Row 2, Column 1: Host Chooser DDL T3: A Drop-Down List component (whose flow action is Update, which permits choosing a new host. The current page updates itself with new information based on the chosen host.)

Preliminary Setup

As with Tutorials 2 and 3, it is a good idea to keep the work you did on Tutorial 3 intact so that you can reuse it if necessary. To keep the tutorials separate, create a new user and copy Tutorial 3 to the new workspace. There are two options for copying, Normal and Deep. By performing a deep copy you copy all the dependent pages, as well as all the other definitions. For a discussion of the copy options, see the *Web Component User Guide*.

To copy the Tutorial 3 module to Tutorial 4:

1. Create a new user (*tutorial4*, *tut0rial*), as you did for Tutorials 2 and 3.
2. Log in to Foglight® as *tutorial4*.
3. On the navigation panel, choose **Configuration > Definitions**.
4. In the Module List pane, choose **Other User Definitions** > **tutorial3**.

IMPORTANT: We assume that the state of Tutorial 3 is what it was after the additional activities were completed.

5 In the Module Contents pane, click the drop-down list, and select **Views**.

6 Select **Configuration** > **Definitions**. and in the Module List pane select **Other User Definitions** > **tutorial3**.

7 In the list of views, select **All Hosts**.

The Definitions pane shows the configuration of the All Hosts view.

8 Click **Copy** and from the menu that appears, choose **Deep**.

9 In the **Copy** dialog box that appears, in the **Select a target module** area, choose **My Definitions** and click **OK**.

Since you are logged in as user *tutorial4*, you now have a separate copy of the module you created in Tutorial 3. You can use this copy to continue your work in Tutorial 4 without affecting the work you did for Tutorial 3.

Defining the Alarms Query

The drilldown page is going to display information about the alarms generated against a selected host, so a query is an obvious choice for getting the required data. However, the data retrieved by the query should be for a specific host, but the host is only chosen at run time as a result of user interaction. The solution is to use a parameterized query. The parameter value is assigned at run-time by selecting a row in the Host Details T3 table.

To configure a query returning Alarms that is passed a Host as a parameter:

1 On the navigation panel, choose **Dashboards** > **Configuration** > **Definitions**.

The Definitions area appears.

2 In the Module Contents pane, click the drop-down list, select **Queries**, and click Add .

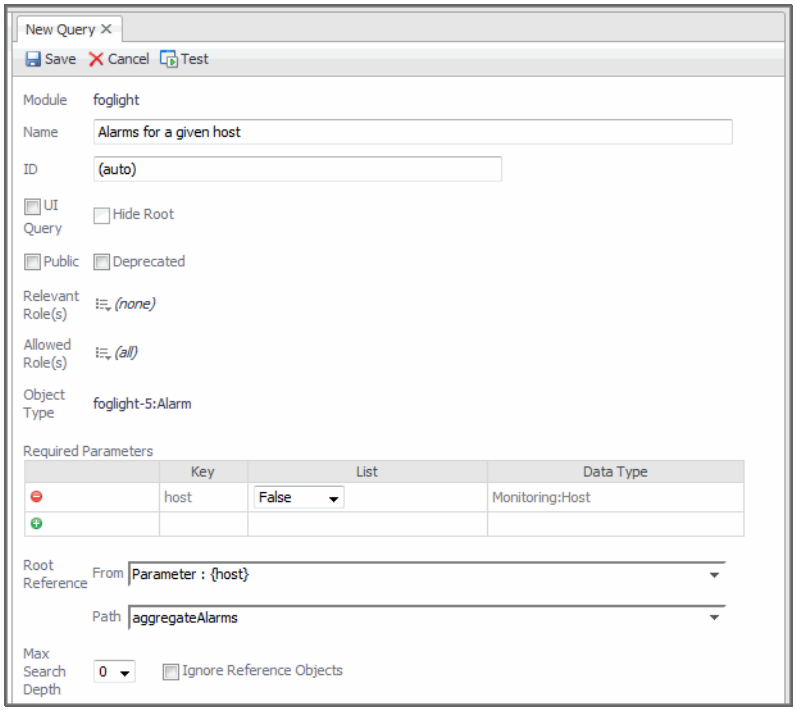
The **Add Query** dialog box appears.

3 In the **Add Query** dialog box, on the **Create a Blank Query** tab, type **Alarm** in the search box, select the **Alarm** row, and click **OK**.

The *Definitions* pane displays the query editor, which contains all the fields that can be used to construct a query. The required fields are marked by an exclamation point icon on the right side of the field.

Configure the query using the information in the following screen:

Figure 1. Configuring a new query



Note that a Root Path of `{host}` is selected. The query requires that a host is passed to it so that it can return only the alarms generated against that host.

At this point we have a valid query.

4 Test the query by clicking the **Test** button, appearing at the top of the query definition.

> **IMPORTANT:** The Test button is not available if you are using Internet Explorer® 6.

No results can appear until you specify a host, which is the required input to the query.

5 In the **Input Values** dialog box that appears, click the button on the right of the **host** box, and choose a host.

In the **Query Results** dialog box that appears, you can see that the query returns a list of objects by expanding any of the any alarms. Some host properties have numerical values, while others, which are themselves objects, do not.

6 Close the **Query Results** dialog box.

7 Click **Save** to save the query.

Configuring a Table of Alarms

In this exercise you build a generic table that lists alarms. When the table is actualized, the data in the table will depend on which host was chosen from the parent dashboard. Choosing a particular host in the parent dashboard is the action that triggers the display of the alarm table.

To configure a table of alarms:

1 On the navigation panel, under **Dashboards**, choose **Configuration > Definitions**.

2 In the Module List pane, select **My Definitions**.

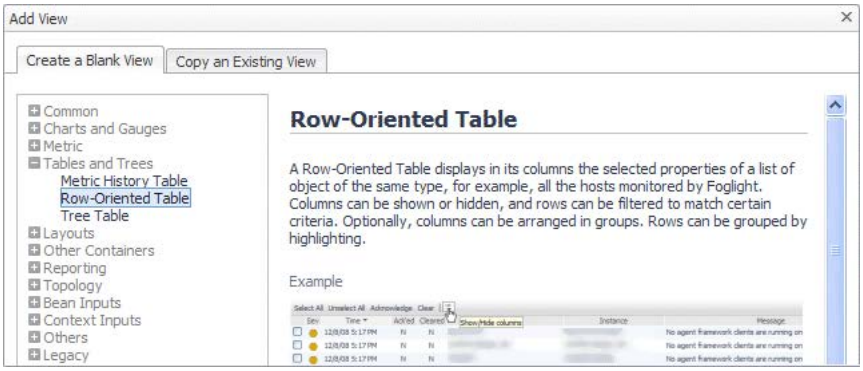
3 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

The **Add View** dialog box appears.

4 In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.

5 On the **Create a Blank View** tab, choose **Tables and Trees > Row-Oriented Table**.

Figure 2. Create a Blank View tab



6 Click **OK**.

7 Populate the values on the view's **General** tab.

a In the **Name** box, type `Alarm Table for Host T3`.

b Leave the **Custom Purpose(s)**, **Relevant Role(s)**, and **Allowed Role(s)** values unchanged.

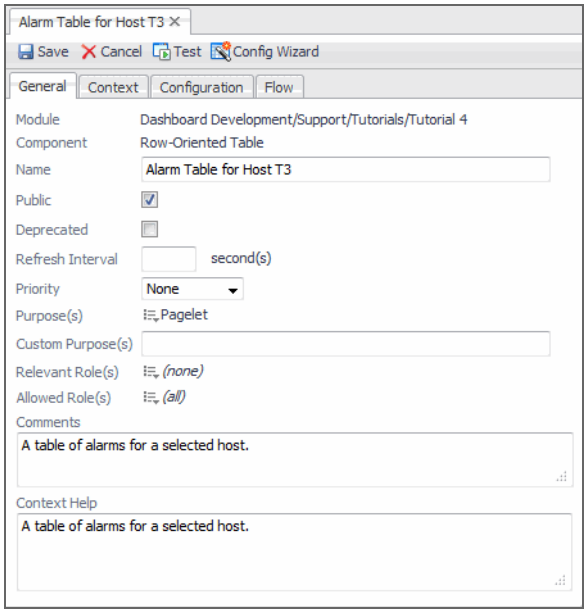
For more information about these settings, see the *Administration and Configuration Help*.

c On the right of **Purpose(s)**, click Edit. In the dwell that appears, select the **Pagelet** check box, and click **Apply**.

d In the **Comments** and **Context Help** boxes, type `A table of alarms for a selected host`.

The table is not fully configured yet, so if you try and save it now you will get an error that informs you that column values have not been set. The next procedure shows you how to choose columns for the table. When configured, the data in the columns' cells is dynamically generated, and depends on the host.

Figure 3. Configuring table settings



Adding columns to the table

To add columns you use the *Configuration* tab, but before you do, define a Context to tell the component that it will require an object of type Host for its operation.

To set the context:

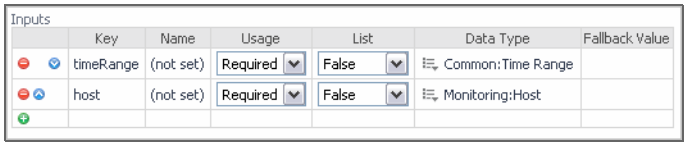
NOTE: This procedure continues from Configuring a Table of Alarms on page 55

1 Open the **Context** tab.

2 Under **Inputs**, click Add Input and click on the new row.

Declare a context using the information in the figure:

Figure 4. Configuring context inputs



This configuration causes the context key `host` to be available to the table. As to the process described in Tutorial 3, you need to configure the **Host Details T3** page to pass `Host` values to the table.

To add a severity column to the table:

NOTE: This procedure continues from Configuring a Table of Alarms on page 55

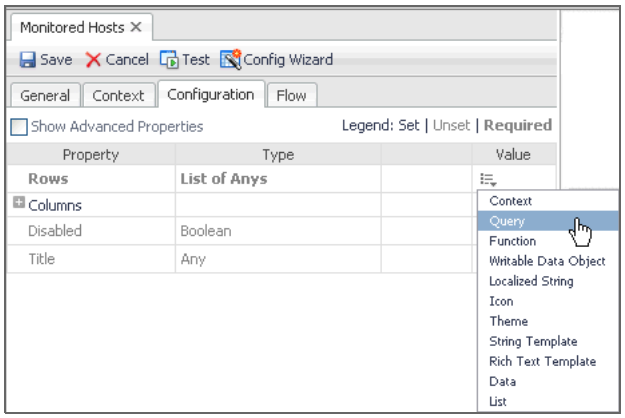
1 Open the **Configuration** tab.

The table designer page appears.

2 We need to instruct the table to populate its rows using a query.

On the **Configuration** tab, in the **Rows** row, in the **Value** column, click Edit and choose **Query** from the list that appears.

Figure 5. Associating a query with a table column



The **Edit - Rows** dialog box appears.

3 In the **Edit - Rows** dialog box, click the **Query** box and choose **My Queries > Alarms for a given host**.

4 In the table, in the **Parameter {host} : of type Monitoring:Host** row, in the **Value** column, click Edit , and in the list that appears, choose **Context**.

5 In the **Edit - Parameter** dialog box that appears, click the **Input Key** box, and choose `host (Host)` from the list that appears.

6 Leave the other values unchanged, and click **Set** to close the **Edit - Parameter** dialog box. In the **Edit - Rows** dialog box, click **Set** to close it and return to the **Configuration** tab.

The table is now configured to retrieve information from a query that returns a list of hosts. We still have to define which property of a host to display in a column. For that, we will have to edit the default column and define its content.

7 On the **Configuration** tab, expand the **Columns** node.

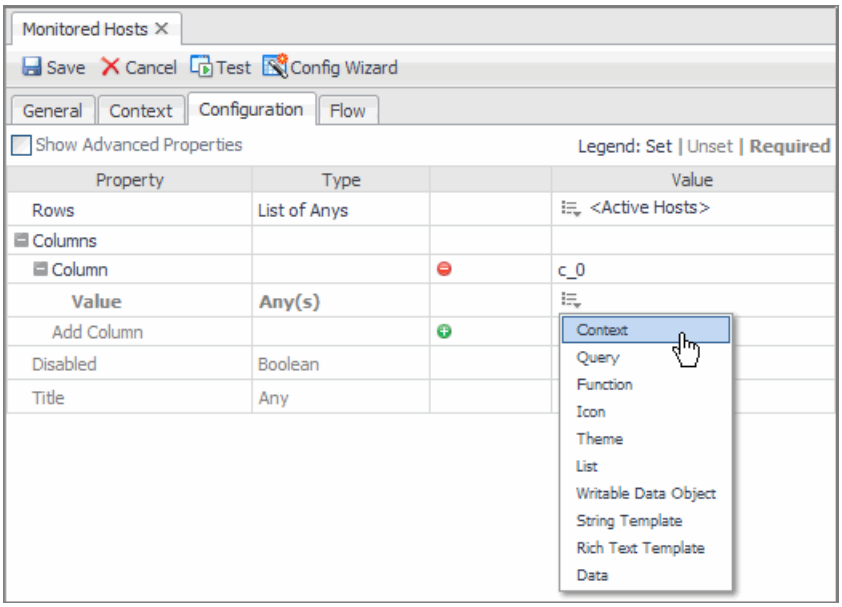
A sub-node called **Column** appears.

8 Expand the **Column** node.

A sub-node called **Value** appears.

9 In the row containing the **Value** node, in the Value column, click Edit , and choose **Context** from the list that appears.

Figure 6. Associating context inputs with a table column



The **Edit - Value** dialog box appears.

10 In the **Edit - Value** dialog box, click the **Input Key** box, and choose **currentRow (Alarm)** from the list that appears.

11 Click the **Path** box and choose **Severity - Alarm Severity** from the list that appears.

This selection places a severity icon in the first column.

12 In the **Edit - Value** dialog box, click **Save**.

Next, we add a column to display alarm messages.

To add a message column:

1 On the **Configuration** tab, ensure that the **Columns** node is expanded, and click to add another column to the table.

Another **Column** node appears, along with its **Value** sub-node.

2 In the second column's **Value** row, in the **Value** column, click Edit , and choose **Context** from the list that appears.

The **Edit - Value** dialog box appears.

3 In the **Edit - Value** dialog box, click the **Input Key** box, and choose **currentRow (Alarm)** from the list that appears.

4 Click the **Path** box and choose **Message - String** from the list that appears.

This choice place an alarm message in the second column.

5 In the **Edit - Value** dialog box, click **Save**.

We illustrate another use of parameters by giving the table a title that contains the name of the host.

To add a parameterized title to the table:

1 On the **Configuration** tab, in the **Title** row, click Edit , and choose **Set to Binding > String Template** from the list that appears.

The **Edit - Title** dialog box appears.

2 In the **Edit - Title** dialog box, in the **Value** box, type `Alarms Table for {0}`.

The `{0}` construct designates a positional parameter. When you use this construct, a row appears in the table below that allows you to associate this parameter with a binding.

3 In the **Parameter {0}** row, in the **Value** column, click Edit , and choose **Context** from the list that appears.

The **Edit - Parameter {0}** dialog box appears.

4 In the **Edit - Parameter {0}** dialog box, click the **Input Key** box, and choose `host (Host)` from the list that appears.

5 Click the **Path** box, and choose `Name - String` from the list that appears.

6 Click **Set** to close the **Edit - Parameter {0}** dialog box.

7 Click **Set** to close the **Edit - Title** dialog box.

8 On the Configuration tab, click Save to save your changes to the **Alarm Table for Host T3** view.

Configuring a Chart Component

Charts can have a large number of properties, but we begin by constructing a simple chart and introduce a more complex example in a later tutorial. For an introduction to Foglight charts, see the *Foglight User Help*.

To add a chart component:

1 On the navigation panel, under **Dashboards**, choose **Configuration > Definitions**.

2 In the Module List pane, select your tutorial module.

3 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

The **Add View** dialog box appears.

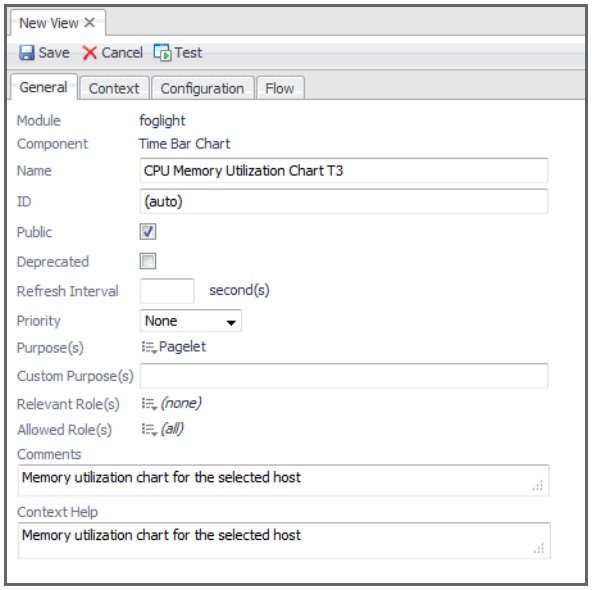
4 In the **Add View** dialog box, on the **Create a Blank View** tab, choose **Charts and Gauges > Time Plot Chart**.

> **IMPORTANT:** A Time Plot Chart is used for showing metric data. If you need to plot specific data, use a Bar or Pie Chart.

5 Click **OK**.

In the Definitions pane, on the **General** tab, populate the settings using the following values.

Figure 7. Chart settings



To set the context:

- Follow the first procedure in Adding columns to the table to make a `Host` object a required input.

To set the chart's data source:

1. Open the **Configuration** tab.

   Observe that this type of chart has as its principal properties a header, footer, time axis and metric data. The time axis by default takes its settings from the current time range, which leaves only the metric data as an essential setting.

2. Expand the **Metric Data - Single Parent** node.

3. In the **Metric Parent** row, in the **Value** column, click Edit , and choose **Context** from the list that appears.

   The **Edit - Metric Parent** dialog box appears.

4. In the **Edit - Metric Parent** dialog box, click the **Input Key** box, and choose `host (Host)` from the list that appears.

5. Click **Set** in the **Edit - Metric Parent** dialog box to close it.

6. On the Configuration tab, under the **Metric Data - Single Parent** node, expand its **Single Metric Bindings** sub-node and then its **Single Metric Binding** sub-node.

7. In the **Metric** row (a sub-node of **Single Metric Binding**), in the **Value** column, click Edit , and choose **Context** from the list that appears.

8. In the **Edit - Metric** dialog box that appears, click the **Input Key** box, and choose `host (Host)` from the list that appears.

9. Click the **Path** box, and choose **Memory - Memory > Utilization - Metric** from the list that appears.

10. Click **Set** to close the **Edit - Metric** dialog box.

11. On the **Configuration** tab, in the **Title** row, in the **Value** column, click Edit , and choose **Set to Binding > String Template** from the list that appears.

12. In the **Edit - Title** dialog box that appears, in the **Value** box, type `Memory Utilization for {0}.`

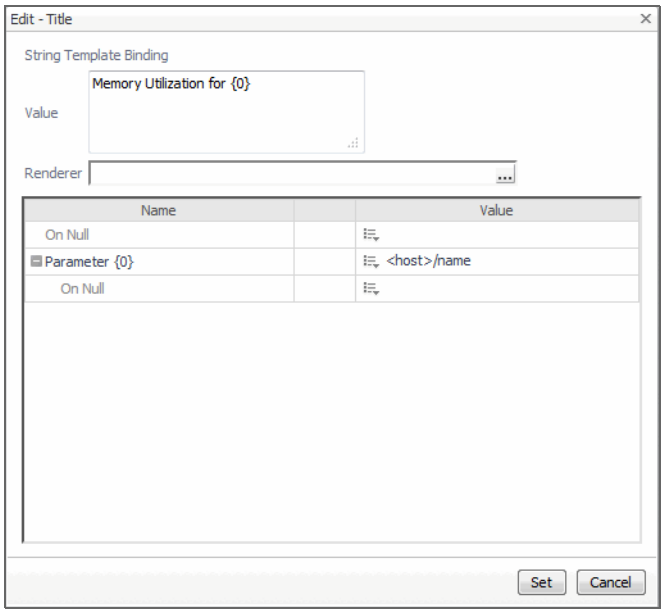
13. In the table below, in the **Parameter {0}** row that appears, in the **Value** column, click Edit , and choose **Context** from the list that appears.

14 In the **Edit - Parameter {0}** dialog box that appears, click the **Input Key** box, and choose `host (Host)` from the list that appears.

15 Click the **Path** box, and choose `Name - String` from the list that appears.

16 Click **Set** in the **Edit - Parameter {0}** dialog box to close it.

Figure 8. Edit - Title dialog box



17 Click **Set** in the **Edit - Title** dialog box to close it.

18 On the **Configuration** tab, click save on the toolbar to save your changes to the **CPU Memory Utilization Chart T3** view.

Configuring a Label with an Action

There are times when you may find it useful to add a link back to a previous page even though there are built-in ways, such as the history list (breadcrumbs) at the top of a page, for accomplishing this task.

We will use a label with an action in this tutorial to show another choice for a flow.

To define a Label component:

1 On the navigation panel, under **Dashboards**, choose **Configuration > Definitions**.

2 In the Module List pane, select your tutorial module.

3 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

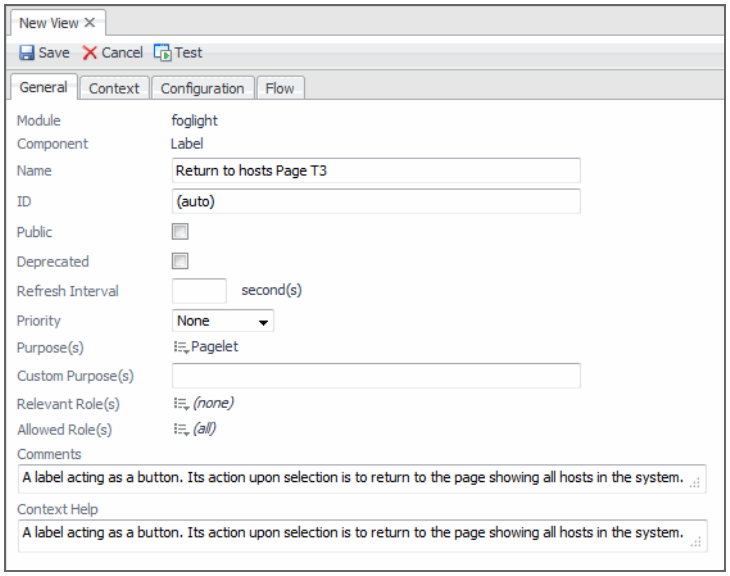
The **Add View** dialog box appears.

4 In the **Add View** dialog box, on the **Create a Blank View** tab, choose **Common > Label**.

5 Click **OK**.

6 In the Definitions pane, on the **General** tab, populate the settings using the following values.

Figure 9. New label component settings



To set the label text:

1. Open the **Configuration** tab.
2. In the **Label** row, in the **Value** column, click Edit , and choose **Set to Value > String** from the list that appears.
3. In the **Edit - Label** dialog box that appears, in the **String Value** box, type `Return to Parent`, and click **Set** to close the dialog box.
4. On the **Configuration** tab, in the **Title** row, in the **Value** column, click Edit , and choose **Set to Binding > String Template** from the list that appears.
5. In the **Edit - Title** dialog box that appears, in the **Value** box, type `Return to Hosts`, and click **Set** to close the dialog box.

> **TIP:** The difference between a String and a String Template is that a String Template can accept parameters. There was no need to use a String Template in this example other than to familiarize you with its dialog box.
>
> The Localized String template should be used if other languages besides English are supported.

When you add this component to the Grid container you can choose to show the title. Otherwise, only the text string defined by the **Label** property appears.

To set a flow action on the label:

1. Open the **Flow** tab.
2. Click the **Selection** row.

   The **Edit - Selection** dialog box appears.
3. In the **Edit - Selection** dialog box, ensure that the **Leave unspecified** check box is cleared.
4. Click the **Flow type drop-down** box and choose **Previous** from the list that appears.

   > **TIP:** By assigning **Previous** to the flow type, you create a link back to the page that invoked the current page.
5. Click **Save** to save your changes and close the **Edit - Selection** dialog box.
6. Click **Save** on the toolbar to save the changes to the Label component.

Configuring the Drilldown Page

Now that you have some component views configured, you need a page to hold them. This section shows you how to create a page and place the required components in it. As we have done before, we use a Grid container.

To create a dependent page called 'Host Details T3':

- Follow the instructions in Tutorial 3, Create a dependent page up to but not including Define a flow, only in this case use `Host Details T3` as the component name when defining the flow, and ensure that you set a `host` context.
- Use `A details page for the selected host` as the component description.

Populate the dependent page

The first component we add is the Time Plot Chart.

To add the Time Plot Chart component to Host Details T3:

1. With the **Host Details T3** view open for editing, open its **Layout** tab.
2. Click **Add**.
3. In the **Add View** dialog box that appears, choose **Select existing view** and click **Next**.
4. Ensure the **Filter By: purpose** and **validity** check boxes are selected.
5. Choose **Dashboard Development > Support > Tutorials > Tutorial 4 > CPU Memory Utilization Chart T3**, and click **Next**.
6. In the refreshed **Add View** dialog, set the width to `400 px` and height to `350 px`.
7. In the **Row** and **Column** boxes, type `0`.

   This places the newly added view in the first cell.
8. Select the **Show title** and **Show border** check boxes.
9. Leave the other settings with their default values.
10. Click **Finish**.

To add the Row-Oriented Table component to Host Details T3:

1. With the Host Details T3 view open for editing, open its **Layout** tab.
2. Click **Add**.
3. In the **Add View** dialog box that appears, choose **Select existing view** and click **Next**.
4. Ensure the **Filter By: purpose** and **validity** check boxes are selected.
5. Choose **My Views > Alarm Table for Host T3**, and click **Next**.
6. In the refreshed **Add View** dialog, set the width to `425 px` and height to `400 px`.
7. In the **Row** box, type `1`, and in the **Column** box, type `0`.

   This places the newly added view immediately under the **CPU Memory Utilization Chart T3** view.
8. Select the **Show title** and **Show border** check boxes.
9. Leave the other default values unchanged.
10. Click **Finish**.

Placing a table under a chart may take up more vertical space than is available on your monitor. Next, to ensure that the label and drop-down list components can be accessed, we add a vertical scroll bar to the **Host Details** T3 page.

To add scrollbars to Host Details T3:

1. With the **Host Details T3** view open for editing, open its **Configuration** tab.
2. On the **Configuration** tab, select the **Show Advanced Properties** check box.
3. In the list of properties, locate and expand the **Page Options** node.
4. In the **Scrollbars** row, in the **Value** column, click Edit , and choose either **Vertical** or **Auto** from the list that appears.

Now add the label and drop-down list components.

To add the Label component to Host Details T3:

1. With the **Host Details T3** view open for editing, open its **Layout** tab.
2. Click **Add**.
3. In the **Add View** dialog box that appears, choose **Select existing view** and click **Next**.
4. Ensure the **Filter By: purpose** and **validity** check boxes are selected.
5. Choose **My Views > Return to Hosts Page T3**, and click **Next**.
6. In the refreshed **Add View** dialog, set the width to `350 px` and height to `Automatic`.
7. In the **Row** box, type `2`, and in the **Column** box, type `0`.

   This places the newly added view immediately under the **Alarm Table for Host T3** table.
8. Select the **Show title** and **Show border** check boxes.
9. Leave the other default values unchanged.
10. Click **Finish**.
11. On the view editor toolbar, click **Save**.

Configuring a Drop-Down List

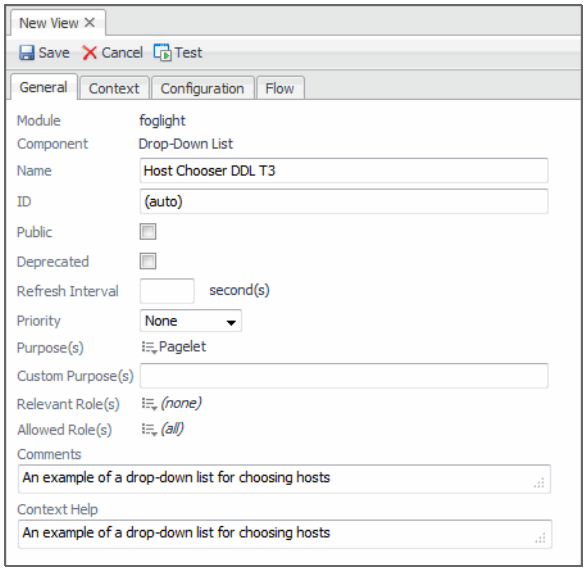
This example illustrates another way of changing the contents of a page. In this case, the page updates with information about another host that you choose from a Drop-Down List component.

To define a Drop-Down List component:

1. On the navigation panel, under **Dashboards**, choose **Configuration > Definitions**.
2. In the Module List pane, select your tutorial module.
3. In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

   The **Add View** dialog box appears.
4. In the **Add View** dialog box, on the **Create a Blank View** tab, choose **Context Inputs > Drop-Down List**.
5. Click **OK**.
6. In the Definitions pane, on the **General** tab, populate the settings using the following values.

Figure 10. New drop-down list component settings



To set the context:

- Ensure a `Host` object a required input. For instructions, see the procedure in Adding columns to the table (under To set the context:).

To set the items for the drop-down list:

1. With the **Host Chooser DDL T3** view open for editing, open its **Configuration** tab.
2. In the **Items** row, in the **Value** column, click Edit , and choose **Query** from the list that appears.

   The **Edit - Items** dialog box appears.
3. In the **Edit - Items** dialog box, click the **Query** box and choose **My Queries > Active Hosts** from the list that appears.
4. Click **Set** to close the **Edit - Items** dialog box.
5. On the **Configuration** tab, select the **Show Advanced Properties** check box.
6. In the **Label** row, in the **Value** column, click Edit , and choose **Set to Binding > Context** from the list that appears.

   The **Edit - Label** dialog box appears.
7. In the **Edit - Label** dialog box, click the **Input Key** box and choose **currentItem** from the list that appears.
8. Click the **Path** box and choose **name** from the list that appears.
9. Click **Save** to close the **Edit - Label** dialog box.

To set the flow for the drop-down list:

1. With the **Host Chooser DDL T3** view open for editing, open its **Flow** tab.
2. Click the **Selection** row.
3. In the **Edit - Selection** dialog box that appears, clear the **Leave unspecified** check box.
4. Click the **Flow type** box and choose **Next Page** from the list that appears.

   When an item in the list is selected, the view will update based on the chosen host.
5. Click the **View** box and choose **My Views > Host Details T3** from the list that appears.
6. Under **Context Mappings**, in the **Selected Item** box, type `host`.

7 Click **Save** to close the **Edit - Selection** dialog box.

8 In the view editor, click **Save** to save your work so far.

To add the drop-down List component to Host Details T3:

1 In the Module List pane, ensure your tutorial module is selected.

1 Open the **Host Details T3** view for editing, and open its **Layout** tab.

2 Click **Add**.

3 In the **Add View** dialog box that appears, choose **Select existing view** and click **Next**.

4 Ensure the **Filter By: purpose** and **validity** check boxes are selected.

5 Choose **My Views > Host Chooser DDL T3**, and click **Next**.

6 In the refreshed **Add View** dialog, set the width to `350 px` and height to `Automatic`.

7 In the **Row** box, type `2`, and in the **Column** box, type `1`.

This places the newly added view on the right of the **Return to hosts Page T3** label.

8 Select the **Show title** and **Show border** check boxes.

9 Leave the other default values unchanged.

10 Click **Finish**.

11 On the view editor toolbar, click **Save**.

Flowing Monitored Hosts to Host Details T3

All that remains is to edit Monitored Hosts to flow to the newly-created dependent page.

To edit Monitored Hosts to flow to Host Details T3:

1 In the Module List pane, ensure your tutorial module is selected.

1 Open the **Monitored Hosts** view for editing, and open its **Flow** tab.

2 On the **Flow** tab, click the **Single-row Selection** row, and in the **View** box, replace **Host Monitor** with **Host Details T3**.

3 Test the Monitored Hosts page by navigating to **Dashboards > My Dashboards > All Hosts**.

4 Click a row in the table to initiate the flow action.

5 The dependent page, **Host Details T3**, appears.

Adding a Customizer

A customizer component is a handy way to add a link from to parent page to open a related view. When you add a customizer to a Portlet view , the link appears at the top right of the component's area.

A customizer is not as useful for top level views with a Page designation, where it shows up in the action panel. If you add a customizer to a Page that is not a top-level view, it is visible only if the view's title bar appears in the container.

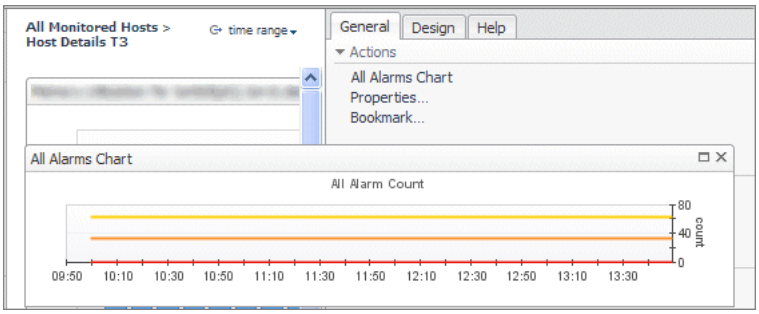
To add a customizer to a page:

1. In the Module List pane, ensure your tutorial module is selected.
2. Open the **Host Details T3** view for editing, and open its **Configuration** tab.
3. On the **Configuration** tab, select the **Show Advanced Properties** check box.
4. In the list of properties, locate and expand the **Customizer** node.
5. Under **Customizer**, in the **View** row, in the **Value** column, click Edit .
6. In the **Edit - View** dialog box that appears, select any view that has `host` set as a **Context Input** key.

   For example, you can choose **Alarms > All Alarms Chart**.
7. Click **Set** to close the **Edit - View** dialog box.
8. On the **Configuration** tab, under **Customizer**, in the **Icon** row, ensure that the **Value** column is set to **None**.
9. In the **Show Title** row, ensure that the **Value** column is set to **true**.
10. In the **Label** row, in the **Value** column, click Edit .
11. In the **Edit - Label** dialog box that appears, in the **String value** box, type `Customizer Test`.
12. In the view editor, click **Save**.

    When you view this page, a link to the view defined in the Customizer appears on the action panel as the first item under **Actions**.

Figure 11. Viewing the newly created page



A Customizer has many useful purposes. It can be used as:

- a filter—for example, an Alarm Filter for alarms tables
- a selector—for example, a Category Selector for Services pages or a Host Selector in a Hosts Table

For views purposed as Dashboard or Page, alternatives to the Customizer are the two choices under **Configuration > Page Options**: **Navigation Panel Views** and **Page Panel Views**, by which you can add views directly to the left and right panels. These choices appear only when the parent page is being viewed.

Adding Nested Views

The **Add** button on the **Layout** tab of a container launches a wizard whose first page contains two choices:

- **Select existing view**
- **Create a nested view**

After choosing **Create a nested view**, the next page of the **Add view** wizard allows you to select a view component just as you did when you chose **Select existing view** in previous examples.

To add a Label component to Host Details T3 as a nested view:

1 Continue editing Host Details T3 in the **Layout** tab.

2 Click **Add** button.

3 In the **Add View wizard** that appears, select **Create a nested view** and click **Next**.

The next page of the **Add View** wizard allows you to choose a label component.

4 Choose **Common > Label**, and click **Next**.

A dialog that contains the context inputs already defined for the container appears with check boxes beside them, allowing you to select any that you want as inputs to your nested view.

5 Select the **timeRange** and **host** check boxes**,** and click **Next**.

This ensures that these context inputs are set without having to configure them in the **Context** tab.

The next page of the **Add View** wizard appears.

6 In the **Row** box, type `0` and in the **Column** box, type `2`.

7 Click the **Vertical Alignment** box and choose **Middle** from the list that appears.

8 Leave the other default values unchanged.

9 Click **Finish**.

The Definitions pane refreshes, showing the nested Label component in the view editor.

10 Open the **Configuration** tab.

11 In the **Label** row, in the **Value** column, click Edit , and choose **Set to Binding > String Template** from the list that appears.

12 In the **Edit - Label** dialog box that appears, in the **String Template Binding** box, type `CPU Utilization for {0}.`

13 In the table below, in the **Parameter {0}** row that appears, in the **Value** column, click Edit , and choose **Context** from the list that appears.

14 In the **Edit - Parameter {0}** dialog box that appears, click the **Input Key** box, and choose **host (Host)** from the list that appears.

15 Click the **Path** box, and choose **Name - String** from the list that appears.

16 Click **Set** to close the **Edit - Parameter {0}** dialog box, then in the **Edit - Label** dialog box, click **Set** to close it.

17 In the view editor, click **Save**.

Summary

In this tutorial you learned how to use parameters in queries and about some new components.

At this point you can begin building more complex designs by consulting the *Web Component Framework*, available from the Foglight® online help.

Additional activities

After completing this tutorial, consider performing the following activities:

- Investigate what effect changing the time range has on the **CPU Memory Utilization Chart: T3** and possibly **Alarm Table for Host T3** views.
- Change the height of the **Alarm Table for Host T3** view by opening the **Host Details T3** view for editing, opening its **Layout** tab, double-clicking the **Alarm Table for Host T3** area, and setting the **Height** property

to **Automatic**. When you chose **Automatic**, you rely on the layout algorithm to adjust the height to a reasonable size, but you can override that by forcing the component to have a specified height.

- The **Alarm Table for Host T3** view can have more entries than the height of the component allows. Add scrollbars to the view and observe the change.

6

Tutorial 5: Using a Grid

This tutorial illustrates some of the more advanced properties of the Grid component. As well, the tutorial outlines how to perform a deep copy of an existing view and how to place a component on the navigation panel.

- Building the Dashboard
- Summary

Before building the dashboard

Before you get started, obtain a deep copy of the **[chart] Disk I/O Utilization** view. Deep copying gives you your own copy of the component and any of its referenced components, which you can edit any way you want. You need to add this view to the grid, and by copying it first you can make changes to the copied version without affecting the installed component.

To deep copy the [chart] Disk I/O Utilization view.

1. On the navigation panel, choose **Dashboards** > **Configuration** > **Definitions**.
2. In the Definitions area that appears, in the Module List pane, choose **Hosts > Host**.
3. In the Module Contents pane, in the drop-down list, ensure that **Views** is selected.
4. Locate and click the row containing the **[chart] Disk I/O Utilization** view.

   The view settings appear in the Definition/Edit pane.
5. In the Definition/Edit pane, click **Copy**, and choose **Deep** from the list that appears.
6. In the **Copy** dialog box that appears, click **My Definitions**, and click **OK**.

Deep copy the **[chart] Disk I/O Utilization** view from **Hosts > Host**.

Modify chart properties

To start off, give the newly copied chart a title and ensure that the legend is enabled.

To edit the chart's Header property:

1. In the Module List pane, click **My Definitions**.
2. Open your copy of the **[chart] Disk I/O Utilization** view for editing.
3. On the **General** tab, clear the **Public** check box, and set the **Priority** to **None**.
4. Open the **Context** tab and ensure the **Usage** column of the host entry is set to **Required**. Like in the previous tutorials, you pass a host object to the component in this tutorial.
5. Configure the **timeRange** context key to use the following settings:
   - **Set Usage**: `Required`
   - **List**: `False`
   - **Data Type**: `Common:Time Range`
6. Open the **Configuration** tab and ensure that **Show Advanced Properties** is selected.
7. Locate and expand the **Header** node.

8 In the **Text** sub-node, in the **Value** column, click Edit ( ), and choose **String Template** from the list that appears.

9 In the **Edit - Text** dialog box that appears, in the **Value** box, type `Disk I/O Utilization for {0}`.

10 In the **Parameter {0}** row that appears in the table below, in the **Value** column, click Edit ( ), and choose **Context** from the list that appears.

> **IMPORTANT:** Only System modules automatically have an associated resource bundle, so localization of the header string is not at issue here. For details on localization and resource bundles, see the *Web Component Guide*.

11 In the **Edit - Text** dialog box that appears, click the **Input Key** box, and choose **host (Host)** from the list that appears.

12 Click the **Path** box, and choose **Name - String** from the list that appears.

13 Click **Set** to close the **Edit - Parameter {0}** dialog box, then in the **Edit - Text** dialog box, click **Set** to close it.

To edit the chart's Legend property:

1 On the **Configuration** tab, locate and expand the **Legend** node.

2 In the **Visible** sub-node, in the **Value** column, click Edit ( ), and in the **Edit - Visible** dialog box that appears, ensure the **Boolean value** check box is selected.

3 Click **Set** to close the dialog box.

4 On the view editor, click **Save**.

These are the only changes you need to make to the chart after copying it from **Hosts > Host**.

Using a list as a chooser

One purpose of this tutorial is to help you investigate the layout properties of the Grid component. Once more, we choose the familiar example of a host. Rather than limiting you to a single host, we use a Drop-Down List to pick any host in your monitored system. Alternatively, we can build a top-level dashboard that lists all hosts (like in Tutorial 2), and then configure a drill down page as a grid displaying host information. This tutorial, however, shows you how to update a top-level page by making a choice within the page and then having all the other views update themselves.

In this example, selecting a host in the drop-down list reloads the page and uses the selected host to set the context values. This is a simple but effective way to use one page to look at the properties of a number of different hosts.

Choosing the views

Because we are interested in Grid properties, we do not need to take the time to build all the views that we want to place in the cells of the Grid component. Instead, we will pre-built views and investigate what shape they take after adding them to the Grid cells.

The views that we will choose are:

- A Row-Oriented Table for choosing hosts. We build this table in the tutorial.
- **Agents Running On A Given Host** from **Hosts > Host Analysis**.
- **CPU Memory Utilization Chart T3** from the module containing the tutorial views.
- **Alarm List with Filter** from the **Alarms**.
- **[chart] Disk I/O Utilization** from the user module **My Definitions** for Tutorial 5.

For the Customizer, we make use of the **Host Browser** view from **Hosts > Host**, inserting it as the Customizer for this page.

Building the Dashboard

We begin by constructing the Grid component's shell. After that, we lay out the contained views. After adding all of the required views, you can experiment with various layout settings to see how they affect the grid's appearance.

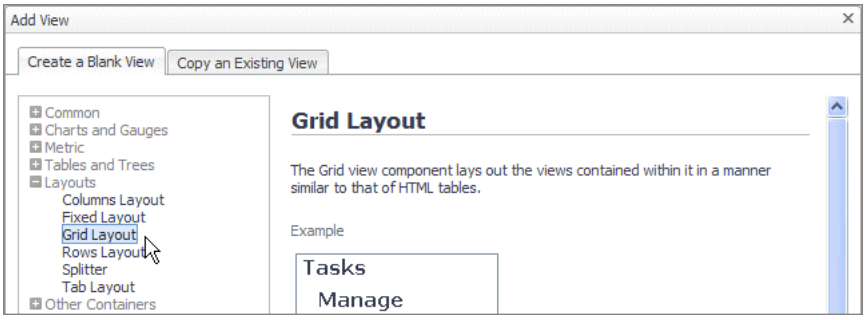
To configure the General page of the dashboard:

1 In the Module List pane, ensure that **My Definitions** is selected.

2 In the Module Contents pane, ensure that **Views** is selected in the drop-down, and then click Add .

3 In the **Add View** dialog box that appears, on the **Create a Blank View** tab, choose **Layouts > Grid**, and click **OK**.

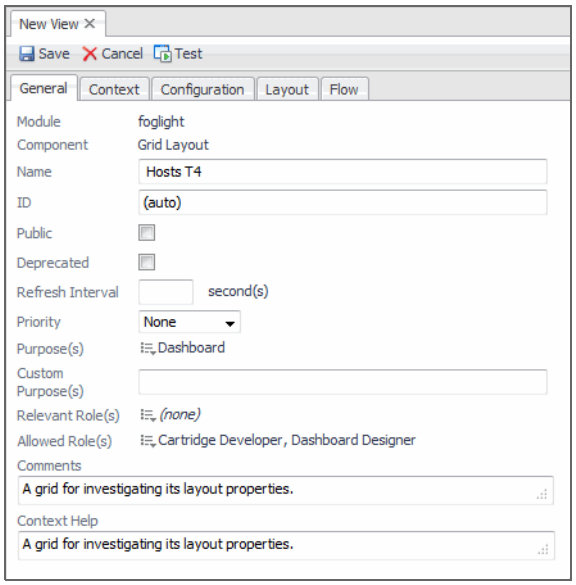
Figure 1. Add View



4 The view appears open for editing in the **Definitions** pane, with the **General** tab open.

5 Configure the settings on the **General** tab using the following values.

Figure 2. Dashboard settings



6 Click **Save** to save your work so far, and then click **Edit** to continue editing this dashboard.

Set a context

Define a Context to instruct the component accept a `Host` object as context input and supply the component with actual host-related data. You configure this dashboard to permit the choice of a different host and then update itself.

To set the context:

1 Open the **Context** tab.

2 On the **Context** tab, under **Inputs**, click Add Input to add a new row to the table.

3 Configure the newly added context key using the following settings:

- **Key**: `host`
- **Name**: Leave unchanged
- **Usage**: `Optional`

> **NOTE:** This dashboard does not have a required `Host` input. You will set an initial value in the context in a following step.
>
> The `host` context key is necessary because you use it to supply the host name and state to the dashboard title. By setting a default value, you supply the dashboard with an initial host value. Without it, the view might show errors until a host is selected and an update action is triggered.

- **List**: `False`
- **Data Type**: `Monitoring:Host`

4 Click **Save**.

To supply an initial host value:

1 Copy the **Active Hosts** query from Tutorial 4.

a With My Definitions selected in the Module List pane, in the Module Contents pane, click the dropdown list and select **Queries**.

b In the Module Contents pane, click Add.

c In the **Add Query** dialog box that appears, open the **Copy an Existing Query** tab, choose **Dashboard Development > Support > Tutorials > Tutorial 4 > Active Hosts**, and click **OK**.

2 Continue editing the **Hosts T4** view, and open the **Context** tab.

3 On the **Context** tab, under **Inputs**, in the **host** row, in the **Fallback Value** column, click Edit , and choose **Query** from the list that appears.

4 In the **Edit - Context Input Entry "host"** dialog box that appears, click the **Query** box, and choose **My Queries** > **Active Hosts** from the list that appears.

5 Select the **Return First Object in List** check box.

6 Set the On Null value in the table below.

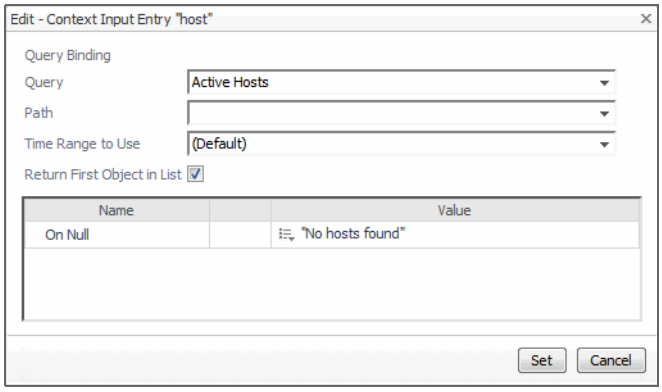
Null values occur when a query fails to return any objects at all. In this case, you are setting a null value in case the query fails to return any objects of the `Host` type.

In the table, in the **On Null** row, in the **Value** column, click Edit , and choose **String Template** from the list that appears.

In the **Edit - On Null** dialog box that appears, in the **Value** box, type `No hosts found`, and click **Set** to close the dialog box.

Ensure that your settings match the following values:

Figure 3. Context input settings



7 Click **Set**.

Adding the views to the dashboard

As in Tutorial 2, the first step here is to give the dashboard a title.

To define the dashboard's title:

1 Open the **Configuration** tab.

2 In the **Title** row, in the **Value** column, click Edit , and choose **Set to Binding > String Template** from the list that appears.

3 In the **Edit - Title** dialog box that appears, in the **Value** box, type `Host: {0} -- State {1}`.

Two parameters appear in the table below: `Parameter {0}` and `Parameter {1}`.

4 For each of these parameters, in the **Value** column, click Edit , and choose **Context** from the list that appears.

5 In the dialog box that appears, choose the following values:

Parameter {0}:

- **Input Key**: `host (Host)`
- **Path**: `Name - String`

Parameter {1}:

- **Input Key**: `host (Host)`
- **Path**: `Aggregate State - Integer`

IMPORTANT: Recall that aggregateState is an integer, but based on its type it has a default renderer that shows it as an icon that reflects the severity of the state.

6 Click **Set** to close the dialog box, and click **Save** in the view editor to save your changes so far.

Adding the views

You need to populate the grid with some existing views. First, we create a host chooser by placing a Drop-Down List in the dashboard.

To modify the Drop-Down List for this tutorial:

1 Replicate the steps you performed to create **Host Chooser DDL T3** in Tutorial 4 (see Configuring a Drop-Down List on page 64), but this time name it **Host Chooser DDL T4**. Do not use the flow settings from Tutorial 4.

2 With the **Host Chooser DDL T4** open for editing, open the **Flow** tab.

3 Click the **Selection** row, and in the **Edit - Selection** dialog box that appears, clear the **Leave unspecified** check box.

The **Edit - Selection** dialog box refreshes, showing additional settings.

4 In the **Edit - Selection** dialog box, click the **Flow type** box, and choose **Update** from the list that appears.

5 In the **Selected Item** box, type `host`.

This flow action updates the page and passes the selected host as a context input, so that data associated with that host appears in the view.

6 Click **Save** to close the **Edit - Selection** dialog box. In the view editor, click **Save** to save your changes to the **Host Chooser DDL T4** view.

To add the Drop-Down List to the dashboard:

1 Open the **Hosts T4** view for editing, and open the **Layout** tab.

2 Click **Add**.

3 In the **Add View** dialog box that appears, select **Select existing view**, and click **Next**.

4 Ensure that the **purpose** and **validity** check boxes are selected.

5 In the refreshed **Add View** dialog box, in the View area, choose **My Views > Host Chooser DDL T4**, and click **Next**.

> **TIP:** If you do not have this component in your current module, deep copy it from the Tutorial 4 module, and then add it to the layout.

6 Set both **Width** and **Height** to `Automatic`.

7 Set both **Row** and **Column** to `0`.

8 Select the **Show Title** and **Show Border** check boxes.

> **TIP:** Checking Show Title is sufficient. The border shows when Show Title is selected.

9 Click **Finish** to complete the layout of the Drop-Down List.

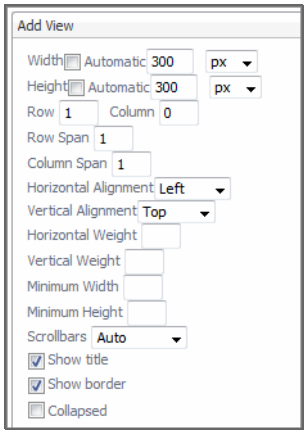
To add the CPU memory utilization chart:

1 In a similar fashion, add the **CPU Memory Utilization Chart T3** view to this dashboard.

> **TIP:** This chart appears listed under **My Views**. In case you do not see it there, ensure the chart is Public to see it as a layout choice.

2 Use the following values to configure the layout settings for this view.

Figure 4. View layout settings

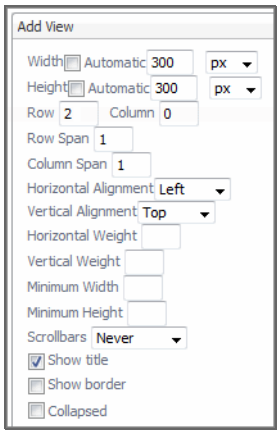


To add an Alarm List with Filter:

- Use the same procedure to add the **Alarm Table for Host T3** view from **My Definitions** using the following values to configure the layout settings for this view.

  TIP: This list appears listed under **My Views**. In case you do not see it there, ensure the list is Public to see it as a layout choice.

Figure 5. View layout settings



The last component to be added is another chart. Rather than building the chart from scratch, copy an existing chart and place the copy in *My Definitions*.

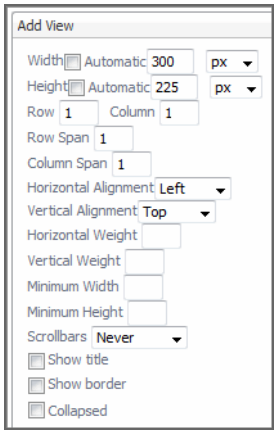
To add a disk I/O utilization chart:

1 By now, you should have a deep copy of the **[chart] Disk I/O Utilization** view from **Hosts > Host**. If you do not, save the grid, make a copy of the chart, and then resume editing the grid.

  TIP: See Before building the dashboard on page 70 for instructions on how to deep copy a view.

2 Add the **[chart] Disk I/O Utilization** view from **My Definitions** using the following values to configure the layout settings for this view.

Figure 6. View layout settings



3 Click **Save** in the view editor to save your changes to the Hosts T4 dashboard.

The Grid is ready for testing.

On the navigation panel, under **Dashboards**, choose **My Dashboards > Hosts T4**.

The Hosts T4 dashboard appears in the display area.

Figure 7. Viewing the newly created dashboard

As a general rule, views in the same grid should have consistent settings. To achieve consistency, you can experiment by changing the settings on one or both components. For more information, see Additional activities on page 78.

Adding a customizer

To add a customizer to the page:

1 Open the **Host T4** view for editing.

2 Open the **Configuration** tab.

3 On the **Configuration** tab, select the **Show Advanced Properties** check box.

4 Locate and expand the **Customizer** node.

5 In its **View** sub-node, in the **Value** column, click Edit .

6 In the **Edit - View** dialog box that appears, choose **Hosts** > **Host Browser** and click **Save**.

7 In the **Icon** row, in the **Value** column, click Edit and choose **None (Default)** from the list that appears.

8 In the **Label** row, in the **Value** column, click Edit .

9 In the **Edit - Label** dialog box that appears, in the **String value** box, type `Customizer Test`, and then click **Set**.

10 Click **Save** in the view editor to save your changes to the Hosts T4 dashboard.

When you open the Hosts T4 dashboard, the customizer link appears on the action panel as the first item in the **Actions** group.

Summary

As shown in the preceding figure, there is quite a bit of unused space on the Hosts T4 dashboard. One way of addressing the vertical height problem is to change the order of the components by placing both charts together on the same row below the table

Another solution is possible. A Columns Layout component can be chosen instead of a Grid Layout. In a Columns Layout, a cell's dimension is independent of its neighbor on either side. Thus, each cell in a column can have the vertical height that it needs without causing spacing problems in adjacent columns.

In other cases, a Rows Layout component may be appropriate. In this component, the width of a cell is not tied to the one above or below

Additional activities

Here are some activities that will deepen your understanding about formatting views in a grid.

- Reposition the components to achieve a better layout. You might want to concentrate on the layout of the alarms table.
- Experiment with weights.
- Experiment with Row and Columns Layout components.

Adjust the layout by fitting cells

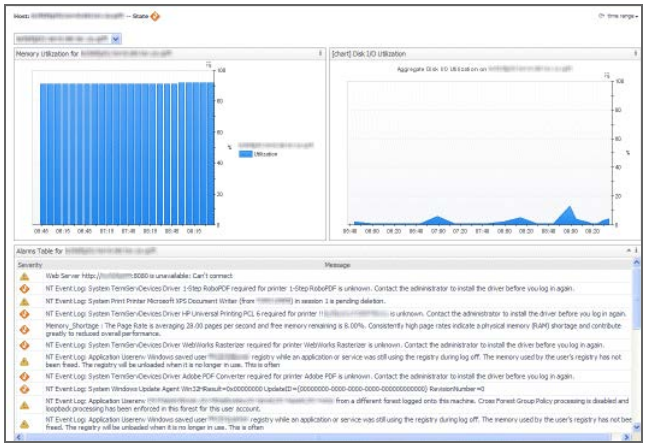
You probably noticed that the layout in this tutorial leaves quite a bit of room for improvement. The **Properties** dialog box for each view in the grid's layout is a good place o start.

Experiment with making these changes to the layout properties:

- Remove the title and border from the **Host Chooser DDL T4** view. There is no need for these items because they take up space needlessly. For future reference, realize that setting a title automatically ensures that a border appears, but setting a border does not automatically cause a title to appear.
- Set the widths of the two charts to **Automatic** and the horizontal alignment to **Fit Cell**. Set the vertical alignment to a fixed number. This causes the charts to align themselves nicely. Later, you can also experiment with weights to see how using simple small numbers adjusts the proportion of space given to one component in comparison to another.
- Set the width of the table to **Automatic**, the **Column Span** to `2`, and **Scroll Bars** to **Auto**.

The following figure gives and indication of the expected result:

Figure 8. Final dashboard



7

Tutorial 6: Reports

By now you should be comfortable with creating and using some of the commonly-used components in the Web Component Framework. We put them to work in this tutorial to show you how to create a report from scratch.

TIP: You can create reports easily using the Reports home page. For instructions, see the Foglight® *User Help*. While the online help provides detailed information about reports and hands-on instructions, this tutorial exposes a deeper level of report generation and is geared toward a technical audience.

There are a number of report examples that are included in the Foglight user interface, but it is likely that as you become more experienced, you want to create your own custom reports that match your exact needs. This tutorial helps you get started.

Along the way, you learn about these additional components: PDF Layout, Page Decoration, and Iterator.

- Configuring a Query on page 81
- Creating a Basic Report Page on page 81
- Creating a More Elaborate Hosts Table on page 84
- Adding a Header on page 89
- Adding a Footer on page 90
- Adding an Iterator for a Multi-Page Report on page 92
- Summary on page 93

Objectives of Tutorial 6

PDF reports are useful for archiving the status of your systems as well as for communicating its recent operational level to interested parties. Whatever your needs, the ability to create informative reports is important. The information needed in a report and the way it should be displayed vary from one enterprise to another, so typically reports are designed on site to meet local needs. This tutorial outlines the way that you can create Foglight® reports using the Web Component Framework.

The objectives of Tutorial 6 are:

- Create a report. This task requires the creation of some context entries to set the title of the report and to retrieve a list of hosts.

  At first you will create a very simple report and then you will add to it.
- Add headers and footers to the report pages.

  Headers and footers are required elements in most reports. You'll see how to add a company logo to the header of a report, and you'll see how to add page numbers to the footer.
- Design the content of the report body and choose the proper view components based on that design.
- Embed a pair of body components in an Iterator so that similar body pages are created simply by iterating a list of hosts.

  The number of active hosts monitored by Foglight is a variable. Using an Iterator permits you to pass a list of hosts resulting from a run-time query and generate a page for each host. Thus, the example serves as a paradigm for any task that performs the same set of actions on objects in a dynamically-generated list.

Designing the page

You begin by constructing the required query. Next, you create a simple report page that contains an existing view. Finally, you create a more complete report that contains a header, a footer, and an iterated view of host data. The design schematic is shown in the table.

Table 1. Report layout for Tutorial 6

Header
Single page: Utilization Summary for All Hosts
Iterated page: System Load Summary. Each page presents a time plot of host statistics and a table of high, low, and average values for selected metrics for the chosen time period.
Footer

Configuring a Query

This tutorial requires a list of hosts, just as in Tutorial 2, but this time we demonstrate a way of accomplishing the objective by using both a query and a context setting. First, you build a query that returns a `HostModel`, and then you will a context setting to select hosts from the `HostModel` object returned by the query.

To build the Host Model T5 query:

1. In the Module Contents pane, click the drop-down list, select **Queries**, and click Add.

   The **Add Query** dialog box appears.

2. In the **Add Query** dialog box, on the **Create a Blank Query** tab, in the **Name** column, click **Host Model** (with **Monitoring** in the **Namespace** column), and then click **OK**.

   The Definitions pane displays the query editor, which contains the information needed to construct a query. You do not have to fill in every box, but the required boxes are marked by an exclamation point icon on the right side of the field.

3. In the **Name** box, type `Host Model T5`.

   You will use this name later on to refer to the query.

4. Leave the **ID** box set to `(auto)`.

5. Ensure that the **UI Query** and **Public** check boxes are clear, **Comments** and **Context Help** empty, and **Relevant** and **Allowed Role(s)** unchanged.

6. Click the **Root Reference** box and choose **Data Sources > Monitoring (Default Instance)** from the list that appears.

7. Most objects are grouped at some level down from the root. In this example, the path to the object we want is in **HostModel > Hosts**. Click the **Path** box and choose **ModelRoots** from the list that appears.

8. Click **Save** to save the query.

Creating a Basic Report Page

Just to show you how it works, create a report page using the PDF Layout component that uses an existing view for its body page. We leave the creation of a header and footer for later.

To create a simple report page:

1. In the Module List pane, ensure that your tutorial module is selected.

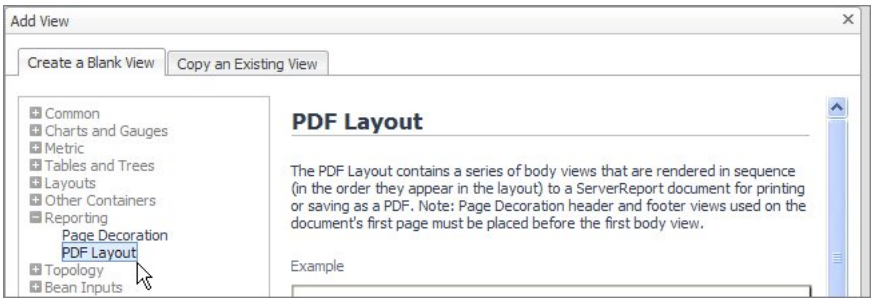
2. In the Module Contents pane, click the drop-down list, select **Views**, and click Add.

The **Add View** dialog box appears.

3 In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.

4 On the **Create a Blank View** tab, choose **Reporting > PDF Layout**.

Figure 1. Create a Blank View tab



5 Click **OK**.

The Definitions pane displays the view editor, which contains all the boxes that can be used to construct a view. The required fields are marked by an exclamation point icon at the right side of the field. In this case, only the **Name** field is required.

6 In the view editor, on the **General** tab, in the **Name** box, type `Hosts Summary Report T5`.

7 Ensure that the **Public** and **Deprecated** check boxes are clear.

8 Leave the **Refresh Interval** box unchanged.

9 Leave the **Priority** set to **None**.

10 On the right of **Purpose**, click Edit , and in the dwell that appears, select the **Report** check box, leaving all of the other boxes clear. Click **Apply**.

11 Leave the **Custom Purpose(s)**, **Relevant Role(s)** and **Allowed Role(s)** unchanged.

12 In each of the **Comments** and **Context Help** boxes, type `A test that displays a simple report page`.

Define a value for the context "*title*":

This context is useful if you embed the report in a container, such as a Grid Layout, and then choose **Show Title** in the layout configuration for the report, which in this case must be purposed as a **Pagelet** as well as a **Report**.

To set the contexts for the report title and to generate a list of host objects:

1 Open the **Context** tab.

2 Under **Inputs**, click Add Input , and set the columns of the newly added row as follows:

- **Key**: `title`
- **Name**: unchanged
- **Usage**: `Optional`
- **List**: `False`
- **Data Type**: `Common:String`

3 In the **Fallback Value** column, click Edit , and choose **String** from the list that appears.

4 In the **Edit - Context Input Entry "title"** dialog box that appears, in the **String value** box, type `Utilization Report - Hosts`, and then click **Set**.

To define a context entry for the Host Model query that returns a single host model:

1 Under **Additional**, click the Add Context Entry, and set the columns of the newly added row as follows:

- **Key**: `hostmodel`
- **Name**: unchanged
- **Evaluate Once**: Selected

2 In the **Value** column, click Edit, and choose **Set to Binding > Query** from the list that appears.

3 In the **Edit - Context Entry "hostmodel"** dialog box that appears, click the **Query** box, and choose **My Queries > Host Model T5**.

4 Select the **Return First Object in List** check box.

5 Click **Set**.

You are about to use this context input to the table of hosts that we construct shortly.

To define a context entry "hosts" for a list of hosts contained in the host model:

1 Under **Additional**, click the Add Context Entry, and set the columns of the newly added row as follows:

- **Key**: `hosts`
- **Name**: unchanged
- **Evaluate Once**: Cleared

2 In the **Value** column, click Edit, and choose **Set to Binding > Context** from the list that appears.

3 In the **Edit - Context Entry "hosts"** dialog box that appears, click the **Input Key** box, and choose **hostmodel (Host Model)** from the list that appears.

4 Click the **Path** box, and choose **Host - List of Hosts** from the list that appears.

5 Click **Set**.

The **hosts** context key now references the list of hosts in the host model.

6 **Optional**. There are two other internally-generated contexts associated with the component: Page Number and Current Time, both of which are optionally available for use in header and footer views. To use these, create context input keys in the child views with the following values:

Page Number

- **Key**: `pageNumber`
- **Data Type**: `Common:Page Number`

Current Time

- **Key**: `currentTime`
- **Data Type**: `Common:Date`

TIP: The PDF Layout component provides two additional special contexts: `pageCount` and `reportTitle`. For more information, see the *Web Component Reference*.

Define a page orientation and title for the report

To define the page orientation and title:

1 Open the **Configuration** tab.

2 In the **Page Orientation** row, in the **Value** column, click Edit (), and choose **Landscape** from the list that appears.

3 In the **Title** row, in the **Value** column, click Edit (), and choose **Set to Binding > Context** from the list that appears.

4 In the **Edit - Title** dialog box that appears, click the **Input Key** box, and select **title (String)** from the list that appears.

   Because you already set this context key to *Utilization Report - Hosts*, that is the title that appears if you embed the report in a container and set its **Show Title** property to `true`.

Adding a sample view to the report

You create pages for the report later on in this tutorial. For now we just want to test basic report construction. There is an existing table in Foglight® that has the same context as needed later on in this tutorial, so we will use it for now.

Next, we add this table to the report.

To add the table to the report:

1 Open the **Views** tab.

2 On the menu bar, click **Add** .

3 In the **Select a View to add to the Report** dialog box that appears, choose an appropriate view under **Hosts > Host**. For example, **Hosts > Host > Host Summary**.

4 Click **Save**.

5 On the **Views** tab, in the row containing the newly-added view, ensure the **Function** column is set to **Body**.

> **TIP:** The other possible choices in this column are **Header** and **Footer**. You add a header and a footer to your report later on in this tutorial.

6 Ensure the **New Page** column is set to **None**.

> **TIP:** The other possible choices in this column are **Before**, **After**, or **Both**. The settings control the placement of page breaks. You can choose to have a page break before, after or both before and after this view. We choose **None** because this is a single page report. There are no other views.

7 The PDF Layout component is now configured. On the view editor toolbar, click Save .

8 Test the report.

9 On the view editor toolbar, click Test PDF .

   To see the report, click the **Test PDF** button (Test PDF) on the toolbar.

10 In the **Input Values** dialog box that appears, change the values for the time range and title. Choose a range during which you know some activity has occurred and then click the **Results** button.

   If you have Acrobat Reader installed, the report appears in a new Web browser window (or tab) you a page containing a table with entries similar to that shown in the figure. The report shows the view you selected in Step 3.

   Even though the report does not include a header or a footer and contains only one view, this exercise helps you become familiar with the basic steps for creating a report.

Creating a More Elaborate Hosts Table

A view like the one you just used serves as a useful introductory item to Foglight® reporting features. This report can be refined to show numerical data supplemented by time plots. This time you build it yourself.

To add a Row-Oriented Table component showing load summaries:

1 In the Module List pane, ensure that your tutorial module is selected.

2 In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

The **Add View** dialog box appears.

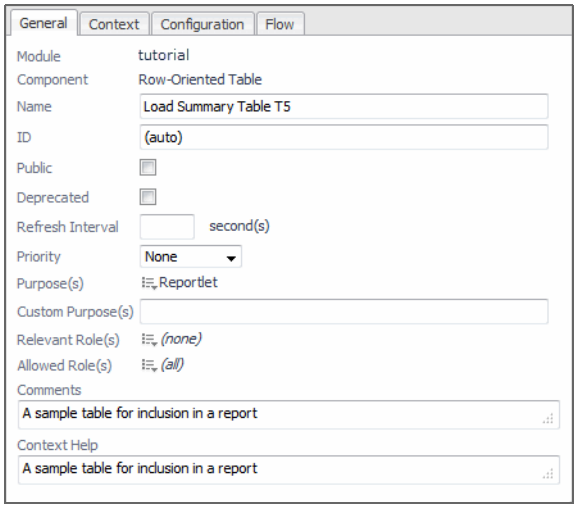
3 In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.

4 On the **Create a Blank View** tab, choose **Tables and Trees > Row-Oriented Table**.

5 Click **OK**.

6 Populate the values on the view's **General** tab as follows:

Figure 2. Row-Oriented Table settings



Configuring the columns for the row-oriented table

This component needs a host context as well, so need to define it.

To set the context:

1 Open the **Context** tab.

2 Under **Inputs**, click Add Input , and set the columns of the newly added row as follows:

- **Key**: `hosts`
- **Name**: unchanged
- **Usage**: `Required`
- **List**: `True`
- **Data Type**: `Monitoring:Host`

3 In the **Fallback Value** column, click Edit , and choose **String** from the list that appears.

4 In the **Edit - Context Input Entry "title"** dialog box that appears, in the **String value** box, type `Utilization Report - Hosts`, and then click **Set**.

Now you need to define the columns for the *Load Summary T5* table.

To define the table's columns:

1 Open the **Configuration** tab.

We need to set each row. This time the rows are populated from the *hosts* context.

2 To tell the table that its rows will be populated from a context selection, locate the row called **Rows**. In the **Value** column, click Edit , and choose **Context** from the list that appears.

3 In the **Edit - Rows** dialog box that appears, click the **Input Key** box, and choose **hosts (List of Hosts)** from the list that appears.

4 Leave all of the other values unchanged, and click **Save**.

The table is now configured to be populated from the `hosts` context that is defined in the **Hosts Summary Report T5** view, but we still have to define which properties of a host appear in the columns. For that, we have to add columns and define their contents.

5 Choose **Columns > Column > Value**.

Next, we set all of the table's columns. We organize them into groups later.

6 In the **Value** column, click Edit , and choose **Context** from the list that appears.

7 In the **Edit - Value** dialog box that appears, click the **Input Key** box, and choose **currentRow (Host)** from the list that appears.

8 Click the **Path** box, and choose **Name - String** from the list that appears.

9 Click **Set**.

10 On the **Configuration** tab, select the **Show Advanced Properties** check box so that you can set the property containing the header name for the newly configured column.

11 Under **Columns > Column > Header**, in the **Value** column, click Edit , and choose **Context** from the list that appears.

12 In the **Edit - Header** dialog box that appears, in the **Value** box, type `Host Name`, and click **Set**.

TIP: By using a Rich Text Template, you can provide localized text for your headers.

13 Under **Columns > Column > ID**, in the **Value** column, click Edit .

14 In the **Edit - ID** dialog box that appears, in the **String value** box, type `Host Name`, and click **Set**.

IMPORTANT: Each entry in the columns that you define must have an ID set so that they can be associated with a group.

Add eight more columns in four groups

At this point you are going to add eight more columns in four groups: CPU, Memory, File System, and Network to complete the definition of the table.

To create the groups:

1 On the **Configuration** tab, expand the **Groups** node and click to add a new group.

2 Under **Groups > Group > Header**, set the group header to `CPU Utilization`.

3 Add three more groups with their **Header** property set to `Memory`, `File System` and `Network`.

To create more columns:

1 On the **Configuration** tab, expand the **Columns** node and click the **Add Column** ( ) button.

2 Choose **Columns > Column > Value**.

3 In the **Value** column, click Edit , and choose **Context** from the list that appears.

4 In the **Edit - Value** dialog box that appears, click the **Input Key** box, and choose **currentRow (Host)** from the list that appears.

5 Click the **Path** box, and choose **CPUs - Host CPUs > Aggregate State Severity - Alarm Severity** from the list that appears.

6 Click **Set**.

7 Under **Columns > Column > ID**, in the **Value** column, click Edit .

8 In the **Edit - ID** dialog box that appears, in the **String value** box, type `CPU Aggregate State`, and click **Set**.

9 Repeat the previous steps to add seven more columns, with the following **ID**, **Path** , and **Renderer** values. Set the **Input Key** value for all of these column to **currentRow (Host)**.

> **IMPORTANT:** Some of the following settings use renderers. Renderers are selected from the Renderer box.

Column 1:

- **ID**: `CPU Average Utilization`
- **Value > Path**: `CPUs > Utilization - Metric > Current - Metric Value > Average - Double`
- **Value > Renderer**: `Management Server > System > Number: No Decimals, With Unit`

Column 2:

- **ID**: `Memory Aggregate State`
- **Value > Path**: `Memory - Memory > Aggregate State - Integer`

Column 3:

- **ID**: `Memory Average Utilization`
- **Value > Path**: `Memory - Memory > Utilization - Metric > Current - Metric Value > Average - Double`
- **Value > Renderer**: `Management Server > System > Number: No Decimals, With Unit`

Column 4:

- **ID**: `Disk Aggregate State`
- **Value > Path**: `Storage - Host Storage > Aggregate State - Integer`

Column 5:

- **ID**: `Disk Average Utilization`
- **Value > Path**: `Storage - Host Storage > Disk Utilization - Metric > Current - Metric Value > Average - Double`
- **Value > Renderer**: `Management Server > System > Number: No Decimals, With Unit`

Column 6:

- **ID**: `Network Aggregate State`
- **Value > Path**: `Network - Host Network > Aggregate State - Integer`
- **Value > Renderer**:

Column 7:

- **ID**: `Network Average Utilization`
- **Value > Path**: `Network - Host Network > Utilization - Metric > Current - Metric Value > Average - Double`
- **Value > Renderer**: `Management Server > System > Number: No Decimals, With Unit`

To add a column to its target group and set the table title and its column order:

1 On the **Configuration** tab, expand the **Groups** node, choose the **Group** node to which you want to add a column, and expand the group's **Column Order** node.

2 Create the following column groups:

Group: CPU Utilization

- Column: CPU Aggregate State

- Column: CPU Average Utilization
- Column: CPU User Time

Group: Memory

- Column: Memory Aggregate State
- Column: Memory Aggregate Utilization

Group: File System

- Column: Disk Aggregate State
- Column: Disk Average Utilization

Group: Network

- Column: Network Aggregate State
- Column: Network Average Utilization

To add a a column to a group, first create a group, and add a **Column ID** node to the group's **Column Order** node. For each column that you want to add to the group, create a separate **Column ID** node, and set its value to the column's **ID** value.

3 Set the table title to `Utilization Summary - Hosts`.

4 In the view editor, click **Save**.

5 Open the **Hosts Summary Report T5** view for editing, and add the newly created **Load Summary Table T5** view to the report, as described in Adding a sample view to the report.

6 With the **Hosts Summary Report T5** view still open for editing, on the **Views** tab, select the **Load Summary Table T5** view and click **Properties**.

7 In the Properties dialog box that appears, on the Configuration Override tab, choose Column/Group Order > Column ID, and ensure it is set to `Host Name`. This causes the **Host Name** to display first.

8 Save the changes to the **Hosts Summary Report T5** view.

Generating the table

Test the work you have done so far by generating a PDF of the table.

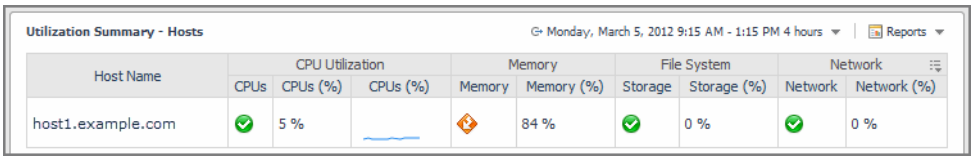
To generate the report:

1 Ensure that **Load Summary Table T5** is open in the Definitions pane.

2 On the Definitions pane toolbar, click Test PDF.

3 Provide the input values, and observe the result.

Figure 3. Newly created able



The table contains nine columns. Having that many columns can present a challenge to the layout algorithm, and it is possible you have noticed some issues that need attention. In the Additional activities section you can learn how to adjust the width of a column to improve the table layout.

Adding a Header

You could attempt to design fixed-layout views to use as the header and footer on a single-page report, but in most cases all you require is a simple one-line header and footer. Multi-page reports would present you with even greater challenges.

The Page Decoration component removes most of the effort in creating headers and footers. It has right, center, and left areas where you can add icons and text. Page numbering is easy because the PDF Layout component passes a special context, `pageNumber`, to views if you assign the *Header* or *Footer* function to them.

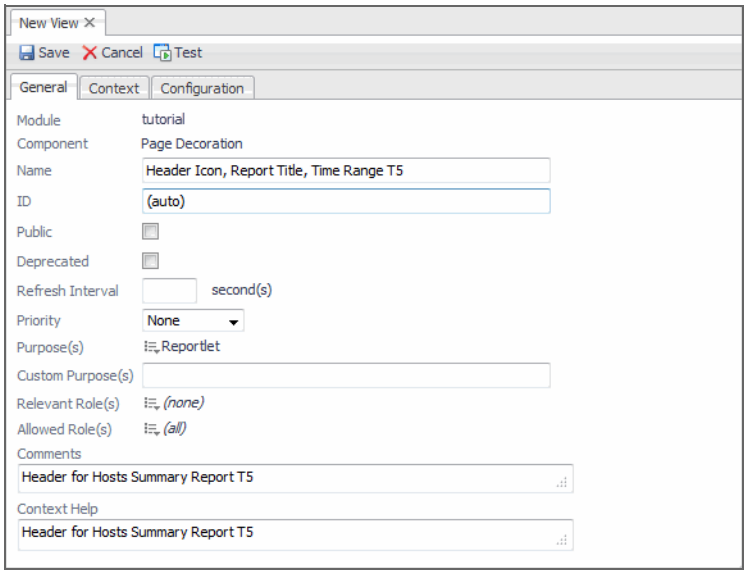
Creating the header component

To create the Header component:

1. In the Module List pane, ensure that your tutorial module is selected.
2. In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

   The **Add View** dialog box appears.
3. In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.
4. On the **Create a Blank View** tab, choose **Reporting > Page Decoration**.
5. Click **OK**.
6. Populate the values on the view's **General** tab as follows:

Figure 4. Header component settings



To set the context:

1. Open the **Context** tab.
2. Under **Inputs**, click Add Input , and set the columns of the newly added row as follows:
   - **Key**: `title`
   - **Name**: unchanged
   - **Usage**: `Required`
   - **List**: `False`

- **Data Type**: `Common:String`

To configure the header:

1. Open the **Configuration** tab.
2. Set the left portion of the header.
   a. In the **Left Image** row, in the **Value** column, click Edit, and choose **Set to Binding > Icon** in the list that appears.
   b. In the **Edit - Left Image** dialog box that appears, click the **Icon** box, choose **Management Server > Reporting > Company Logo** from the list that appears, and click **Set**.

   Note: For instructions on how to add your company logo to this repository, see the *Web Component Guide.*
3. Set the center portion of the header.
   a. In the **Center** row, in the **Value** column, click Edit, and choose **Context** from the list that appears.
   b. In the **Edit - Center** dialog box that appears, click the **Input Key** box, choose **title (String)** from the list that appears, and click **Set**.
4. Set the right portion of the header.
   a. In the **Right** row, in the **Value** column, click Edit, and choose **Context** from the list that appears.
   b. In the **Edit - Center** dialog box that appears, click the **Input Key** box, and choose **timeRange (Time Range)** from the list that appears.
   c. Click the **Renderer** box, and in the **Select a Renderer** dialog box that appears, choose **Management Server > Reporting > Absolute Time Range**, and then click **Save**.
   d. In the **Edit - Right** dialog box, click **Set**.
5. In the view editor, click **Save**.

Adding the header to the report

To add the Header component to the report:

1. Open the **Hosts Summary Report T5** view for editing, and open its **Views** tab.
2. Click **Add** icon.
3. In the **Select a View to add to the Report** dialog box that appears, in the **Name** column, choose **My Views > Header: Icon, Report Title, Time Range T5**, and click **Save**.
4. On the **Views** tab, in the newly added **Header: Icon, Report Title, Time Range T5** row, click the **Function** column, and choose **Header** from the list that appears.
5. Select the **Header: Icon, Report Title, Time Range T5** row and click **Move Up** two times to move this view to the top of the container view.
6. In the view editor, click **Save**.

Adding a Footer

You learned how to add items to all three places in the header, and the footer component is simpler. To demonstrate how to use the PDF Layout's *pageNumber* built-in context, you will place a page number at the bottom-center of the page.

To create the Footer component:

- Follow the procedure for adding a header (see Creating the header component on page 89), making these changes.
  - On the **General** tab, set the **Name** to `Footer: Page Number T5`.
  - Set the **Comments** and **Context Help** to `Footer for Hosts Summary Report T5.`

To set the context:

NOTE: The PDF Layout page needs to pass the pageNumber context to the footer page, so perform these context settings in both Hosts Summary Report T5 and Footer: Page Number T5.

1. Open the **Context** tab.
2. Under **Inputs**, click Add Input, and set the columns of the newly added row as follows:
   - **Key**: `pageNumber`
   - **Name**: unchanged
   - **Usage**: `Required`
   - **List**: `False`
   - **Data Type**: `Common:Integer`

Setting the center portion of the footer

To add a page number to the footer:

1. Open the **Configuration** tab.
2. In the **Center** row, in the **Value** column, click Edit, and choose String Template from the list that appears.
3. In the **Edit - Center** dialog box that appears, in the Value box, type `- {0} -`.

   The table at the bottom refreshes, showing a line for `Parameter {0}`. By using a parameterized entry for the page number you can add text decorations around it.
4. In the **Parameter {0}** line, in the **Value** column, click Edit, and choose **Context** from the list that appears.
5. In the **Edit - Parameter {0}** dialog box that appears, click the **Input Key** box, choose **pageNumber(Integer)** from the list that appears, and click **Set** to close the **Edit - Parameter {0}** dialog box.
6. In the **Edit - Center** dialog box, click **Set** to close it.
7. In the view editor, click **Save**.

Add the footer to the report

To add the Footer component to the report:

1. Open the **Hosts Summary Report T5** view for editing, and open its **Views** tab.
2. Click **Add** icon.
3. In the **Select a View to add to the Report** dialog box that appears, in the **Name** column, choose **My Views > Footer: Page Number T5**, and click **Save**.
4. On the **Views** tab, in the newly added **Footer: Page Number T5** row, click the **Function** column, and choose **Footer** from the list that appears.
5. Select the **Header: Icon, Report Title, Time Range T5** row and click **Move Up** two times to move this view immediately below the **Header: Icon, Report Title, Time Range T5** view.
6. In the view editor, click **Save**.

Test the report

Test the report by clicking the **Test PDF** (Test PDF) button on in the view editor when you display the **Hosts Summary Report T5** view in the Definitions area. Your Web browser displays a page with a header, a footer, Hosts Table - for Given Hosts (assuming you left this page in the report), and a table of host statistics.

Adding an Iterator for a Multi-Page Report

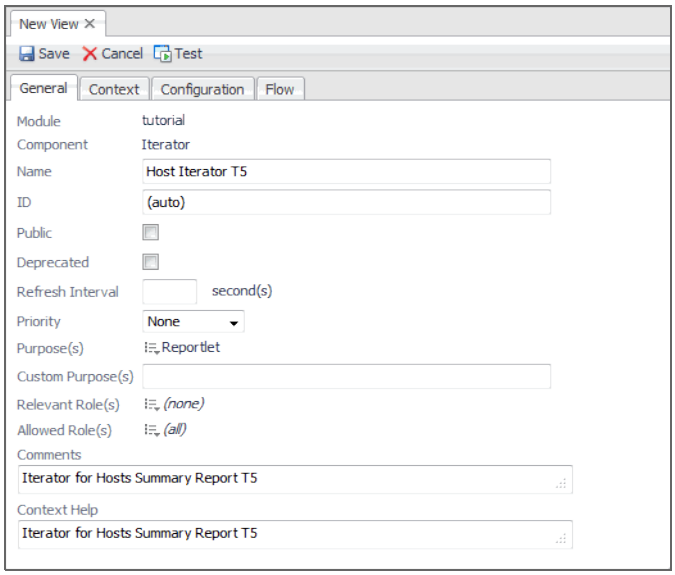
The Iterator component can be used to address a need for creating similar views, such as a report containing detailed information for each host.

To create an Iterator component:

1. In the Module List pane, ensure that your tutorial module is selected.
2. In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

   The **Add View** dialog box appears.
3. In the **Add View** dialog box, ensure that the **Create a Blank View** tab is selected.
4. On the **Create a Blank View** tab, choose **Containers > Iterator**.
5. Click **OK**.
6. Populate the values on the view's **General** tab as follows:

   Figure 5. Iterator component settings



To set the context:

1. Open the **Context** tab.
2. Under **Inputs**, click Add Input , and set the columns of the newly added row as follows:
   - **Key**: `hosts`
   - **Name**: unchanged
   - **Usage**: `Required`

- **List**: `True`
- **Data Type**: `Monitoring:Host`

To set the objects for the iterator:

1. Open the **Configuration** tab.
2. In the **Key** row, in the **Value** column, click Edit .
3. In the **Edit - Key** dialog box that appears, in the **String value** box, type `host`, and click **Set** to close the dialog box.
4. In the **Objects** row, in the **Value** column, click Edit , and choose **Context** from the list that appears.
5. In the **Edit - Objects** dialog box that appears, click the **Input Key** box, choose **hosts (List of Hosts)** from the list that appears, and click **Set** to close the dialog box.

Choosing the Iterated View

You can choose or construct any view whose inputs are host objects, since that is a context set in the **Hosts Summary Report T5** view. For this tutorial, we choose *[table] Memory hogs* from the System module **Hosts > Host**. Any other view whose inputs are acceptable would do as well.

To set the iterated view:

1. On the **Configuration** tab, expand the **Iterated View** node.
2. In the **View** row, in the **Value** column, click Edit .
3. In the **Edit - View** dialog box that appears, choose **Hosts > Host > [table] Memory hogs**, and click **Set** to close the dialog box.

   We are adding an existing view because it is quicker than building a special view only for the purpose of this tutorial. The objective here is to introduce the **Iterator** component.
4. In the view editor, click **Save**.

Adding the Iterator to the report

To add the Iterator component to the report:

1. Open the **Hosts Summary Report T5** view for editing, and open its **Views** tab.
2. Click **Add** icon.
3. In the **Select a View to add to the Report** dialog box that appears, in the **Name** column, choose **My Views > Host Iterator T5**, and click **Save**.

   No other changes are needed because the **Body** page is the default function for this view.
4. In the view editor, click **Save**.

Summary

In this tutorial you learned how to prepare a report with a header and a footer. You also learned how to use an Iterator component to generate multiple views based on a list, thus creating a multi-page report. You viewed the result by manually choosing the **View as PDF** option, and you may be wondering about alternative methods for creating and viewing reports.

Additional activities

In this section you see how to adjust the width of a selected column and how to ensure that the report has a title. Also, you place a report in another component, and in doing so you see how the PDF Layout's `title` property comes into play. You also see what is probably the best method for generating reports automatically by using a schedule.

The final activity provides you with some information on changing the headers and footers in case you want to change them after the first page, or at the beginning of each section of a long report.

Adjusting the table

In most cases the report generator does the best job it can in sizing the various components so that they all fit nicely on the page. Having said that, the generator may have difficulty with some combination of fixed- and variable-width components.

To make the table a little wider we add another column for average utilization of the CPU and we use a sparkline renderer to show the result.

To create a sparkline:

1. In the Module List pane, ensure that your tutorial module is selected.
2. In the Module Contents pane, click the drop-down list, select **Renderers**, and click Add .

   The **Add Renderer** dialog box appears.
3. In the **Add Renderer** dialog box, ensure that the **Create a Blank Renderer** tab is selected.
4. On the **Create a Blank Renderer** tab, choose **Sparkline**, and click **OK**.
5. Click **OK**.
6. In the Definitions area, on the **General** tab, in the **Name** box, type `Sparkline for Report`.
7. In the view editor, click **Save**.

To add a column containing a sparkline to the table:

1. Open the **Load Summary Table T5** for editing, and open its **Configuration** tab.
2. Expand the **Columns** node and add a column with the following settings:
   - **Value**: `Context`

     Input Key: `currentRow (Host)`

     Path: `CPUs - Host CPUs > Percent User Time - Metric`

     Renderer: `My Renderers > Sparkline for Report`
   - **ID**: `CPU User Time`
3. Add the column to the **CPU Utilization** group.
4. Save your changes and test the result.

If you continue to add spark lines to the other groups, you may find that there are alignment problems with the table. In this case the problem seems to be that the Sparkline renderer may request too much space, causing problems for some other columns. In most cases you can cure this type of layout issue by setting the width of some of the variable-width columns.

To set the element width of the Sparkline For Report:

1. Open the **Load Summary Table T5** for editing, and open its **Configuration** tab.
2. Select the **Show Advanced Properties** check box.
3. Expand the column node whose **ID** is set to **CPU User Time**.

4 Expand the column's **Width** node. In the **Sizing** row, in the **Value** column, click Edit , and choose **Specified** from the list that appears.

5 In the **Value** row, in the **Value** column, click Edit .

6 In the **Edit - Value** dialog box that appears, in the **Integer value** box, type `70`, and click **Set**.

7 Save your changes and test the result.

In other situations you may have to set the element widths of other variable-width components, but making just these changes produces a properly-formatted table when it tested during the construction of this tutorial.

Where did the table title go?

If you follow exactly the above procedures , you may notice that even though you set the table's title to `Utilization Summary - Hosts`, appear when you generated the PDF. The situation is quite similar to the point raised in Additional Activities in Tutorial 3. Causing a title to appear is a two-step process, and defining the title's content is only the first step. You also have to configure the parent component to show the title.

To display the table title:

1 Open the **Hosts Summary Report T5** for editing, and open its **Views** tab.

2 Select the **Load Summary Table T5** row and click **Properties**.

3 In the **Properties** dialog box that appears, on the General tab, select the **Show title** check box, and click **Save**.

4 Check **Show Title** and then click **Save**.

5 Save your changes and test the result.

The table title appears in the report.

View the report in a dashboard

Edit the **PDF Layout** component and add an additional purpose by selecting its **Pagelet** check box, and then add it to row `0` and column `0` of a Grid component purposed as a Dashboard.

As part of adding the PDF Layout to the grid, add *title* and *pageNumber* as optional context inputs and give them default values. View the report and note that the title you set in the PDF Layout's Additional Context is displayed.

Schedule a report in Foglight

When the construction of the Foglight® report is finished and it is giving you the result you want, you can schedule a report to run at any time. Creating a report schedule is a two-step process.

First, create a schedule for performing an action that as yet has not been specified.

To generate a report using the Reports dashboard:

1 On the navigation panel, under **Dashboards**, click **Reports**.

The **Reports** page appears in the display area.

2 On the Reports page, click **Run a Report**.

The **Run Report** dialog box appears.

3 In the **Run Report** dialog box, on the **All Templates** tab, in the **Templates** column, locate and select the **Host Summary Report T5**, and click **Next**.

The **Run Report** dialog box refreshes, allowing you to supply context input values defined in the PDF Layout page.

4 Set representative values for these context inputs or accept the default values, but you must set a page number, and click **Next**.

5 Give the new report a name, and click **Next**, then click **Finish**.

The report appears in your Web browser. For more information about reports in Foglight, see the *Foglight User Help*.

Sequential headers and footers

The report you create in this tutorial has a host summary page and then a number of pages, one for each host. In a case like this, or if for any other reason you want to have the first page of the report to have a different header than the rest of the pages, there is a solution.

A report can have multiple headers. The reason you placed the header and footer as the first and second component in the PDF Layout's *Views* tab was to have them registered by the report generator before any body views. Perhaps the table of hosts in your environment spans more than one page. If the footer is placed after the table component, it does not appear until after the table. In fact this is true even if the table contains only a few rows. The report generator delays emitting the footer *until the page after the table is drawn*. In the first part of this exercise, there is no page after the first, so the footer does not appear at all.

Therefore, to switch headers or footers, place the new header or footer component *before* the view that should begin the new page. If necessary, emit a page break between the header and the new component. Using this technique, it is possible to have different headers and footers in all sections of a long report.

Normally, you choose your first body view to fit on page one. But what if you want a new header on the second page and the first page must contain a table or other view that may be several pages long? The first header encountered controls the first page, but if the first body view is several pages long, the header or footer does not change until the whole component is written out. The solution may be to place a thin, empty component after the first page's header and footer, and then place the next header and footer before the multi-page component. The system places the first header and footer on the first page, emit the empty component, set the header and footer for the next page, and begin emitting the multi-page component. This causes the second and following pages to have the header and footer you want.

Try adding a header for the individual host pages. Place this header component just before the iterator in the PDF Layout **Views** tab.

8

Tutorial 7: Creating a Form

This tutorial shows you how to create a form and populate it with data. For the purpose of this tutorial, we use a Form Dialog as a form because this component contains a number of input components and as such can behave like a form. When completed, the end user fills in various input values on the form. In this tutorial, the form fields are placed in a Grid Layout, and the Grid Layout is placed in a Form Dialog container, and clicking the Form Dialog's Submit button transmits this data to the server. The Form Dialog that we use in this tutorial is called *Form for adding a new company manager*. The Grid Layout that contains the input controls is called *Bean Data Fields for Manager*. The Grid has only a few fields, but it illustrates the general idea. A form can have any number of input fields. The dialog supplies the button that initiates the submit action.

- About this Tutorial
- Defining the Enumerated Types for Manager and Staff
- Defining the Manager and Staff Types
- Defining the Company Type
- Defining the Functions for Adding and Removing a Manager
- Defining the Labels
- Defining the Bean Input Components
- Defining the Grid Layout
- Defining the Dialog
- Defining the Table of Managers
- Summary

About this Tutorial

This tutorial covers the following topics:

- **Using the Dialog component as a form**: The Form Dialog component can be used to perform a batch submit of its data. When used in this way, it is normally referred to as a form.
- **Defining Web Component User Interface Types**: If you are building a complex user interface, you often discover the need to create temporary objects to manage the visual components you are creating. You may need to capture input from users and convert it later into more persistent objects, or to remember choices that a user previously made in order to streamline the user's workflow in the future.

  The Web Component Framework allows for the creation of types that are specific to the browser interface mechanism and associated with the modules that contain the rest of your browser interface artifacts.

  Type entities allow you to define types and their properties that can be used in various ways in your interface.
- **Using Bean Input components**: The Web Component Framework provides a mechanism for collecting user data. The collected data can be stored in context keys and passed from the form that collects them to other views, but for many data fields a much more efficient solution exists by using Bean Input components.
- **Storing or removing the collected data using Groovy function scripts**: This tutorial shows you how to create scripts that persist the collected data, but it assumes that you have a working knowledge of Foglight

API and that you are familiar with Groovy. It makes no attempt to be a tutorial for the Foglight API or for Groovy.

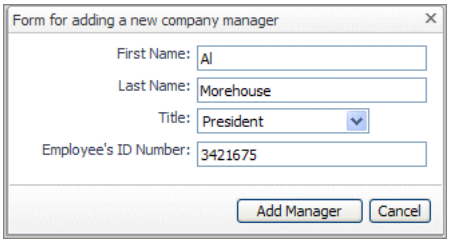
- **Using a function to control the state of a button**: In addition to the four main objectives, an additional activity demonstrates a simple procedure for controlling a button's `Disabled` property. The only parameter passed to the function is a number, which is generated by a Context binding that returns the count of all managers by setting the return type to the context's `Count All` property.

Anatomy of a form and its design objective

The basic elements of a form are:

- A Form Dialog component that holds the rest of the form elements. In most cases the Form Dialog contains a grid which in turn lays out the rest of the form elements. To act as a form, the Form Dialog has its `Form` property set to `true`.
- A Grid Layout that holds the form's input components.
- Input components placed in the grid. The end user fills in these input components, but rather than firing an action on each input component, the Form Dialog defines a submit action that sends the values of all the input components in one operation. Context keys can be used for a small number of fields, but Bean Input components are much more efficient, especially when there are a large number of fields.

Figure 1. Sample input components



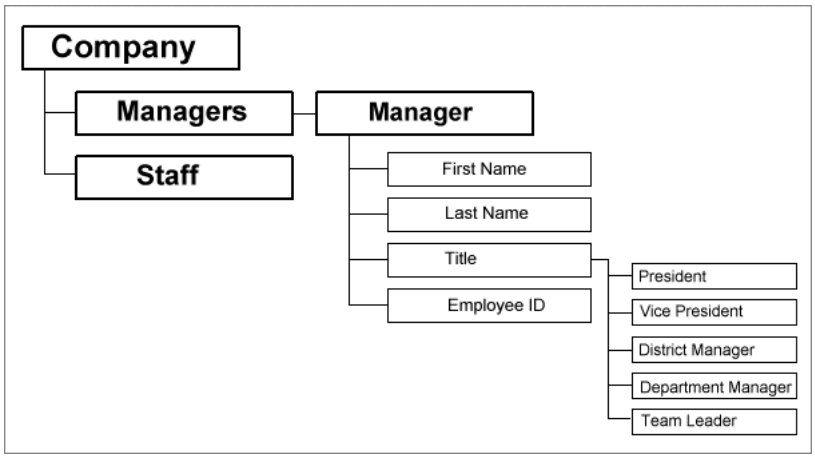
The design objective for the tutorial is to build a data collection utility to hold information about employees. In this fictitious and highly-simplified scenario, one view of a company divides its personnel into managers and staff. The reason for doing this in the tutorial is to have a minimal hierarchy to work with. Thus, the new Data Type to be created in **Types** will look like this:

NOTE: Data Types are described in the Web Component Guide. See the section on Module Types.

Company

- Managers (list of Manager)
  - First Name (String)
  - Last Name (String)
  - Title (Enumeration: Manager Title, another new Module Type)
  - Employee ID (Integer)
- Staff Members (list of Staff)
  - ... (similar properties, but these will not be used in this tutorial)

Figure 2. Sample hierarchy



This model of a company's organization is sufficient for the purposes of this tutorial. No attempt is made to capture the way that the company is organized into functional units because that would add unnecessary complexity to the tutorial.

The Web Component Framework imposes a bottom up approach when constructing its entities. If you create the dependent components first, they are available to be referenced by objects at a higher level. That is why, for instance, the enumerated types Manager Title and Staff Title need to be created before creating Manager and Staff types.

Overview for creating the components

The following is a brief overview of the steps that we take to build this tutorial. Along with listing the necessary components, pointers indicate the actions for collecting, displaying, and storing the user's input.

- Create the enumerated types `Manager Title` and `Staff Title`, with `Common: Enumeration` as their Super Type. For more information, see Defining the Enumerated Types for Manager and Staff on page 100.
- Create the `Manager` and `Staff` types, each consisting of a `First Name`, `Last Name`, `Title`, and `Employee ID`. For more information, see Defining the Manager and Staff Types on page 101.
- Create a `Company` type consisting of a `Name`, list of `Managers` and list of `Staff Members`. For more information, see Defining the Company Type on page 102.
- Create the functions to add and remove items from the list of managers. For more information, see Defining the Functions for Adding and Removing a Manager on page 103.
  - Create the `Add Manager` function.
  - Create the `Remove Manager` function.
- Create labels for Bean Input fields. For more information, see Defining the Labels on page 104.
- Create the Bean Input components. For more information, see Defining the Bean Input Components on page 104.

  A Bean is set using its `Path to Value` property. A `manager` context key is used to set this path.
- Create the Grid Layout (**Bean Data Fields for Manager**) for the manager data fields. For more information, see Defining the Grid Layout on page 106.
- Create the add manager Form Dialog (**Form for adding a new company manager**) to hold the **Bean Data Fields for Manager** grid. For more information, see Defining the Dialog on page 108.

  Here is where you set an additional context key (`manager`) using Writable Data Object to create a new instance of a `Manager` data object whose storage property is `none`. This transitory object holds the data fields set in the dialog. A new `Manager` object is created each time the dialog box appears. Because the storage type is `none`, the `Manager` object must be copied into the `company` structure to persist it, which is accomplished by calling `company.save()` in the `Add Manager` function.

The defined Action, whose flow type is Previous, calls the `Add Manager` function, which requires two parameters:

- `company`, of type `Company`, from context key `company`
- `newManager`, of type `Manager`, from context key `manager`

The function creates a new manager entry and saves it, and the Previous flow (with Update checked) updates the table.

- Create **The Company's Management Team** table to display the added managers. For more information, see Defining the Table of Managers on page 109.

  Here, you configure Actions and Row(s) Selection Actions.

  - The `Add Manager` action displays the **Form for adding a new company manager** as a modal dialog.
  - The `Remove Manager` row selection action is used to update the table after selecting a row and calling the `Remove Manager` function, which requires two parameters: context keys for the selected row and a reference to the `company` object.

- Create a **Company Personnel** Grid Layout to hold the table. For more information, see Defining the Top-Level Dashboard on page 112.

  It is in the **Additional Context** of this component that you use a Writeable Data Object to create the `company` object which creates a null `manager` object. The storage type for the `company` object is `user`, which means each different logon name has a different `company` object that is kept until the server is restarted. See the *Web Component Guide* for information about the available storage mechanism choices.

Next, the following sections contain screen captures that show you all the information you need to construct the types, functions and views for creating a functioning forms example.

Defining the Enumerated Types for Manager and Staff

To create the enumerated types Manager Title and Staff Title:

1 In the Module Contents pane, click the drop-down list, select **Types**, and click Add.

> **IMPORTANT:** Types must be defined in a system module. They are not available in User modules.

The **Add Type** dialog box appears.

2 Create two types using the information provided below. Both types must have `Common:Enumeration` as their Super Type.

The first figure displays the definition of the `Manager Title` type in edit mode and the second figure displays `Staff Title` after it is saved. We will not be using `Staff Title` in this tutorial—it is only there as a placeholder.

Figure 3. Manager Title type

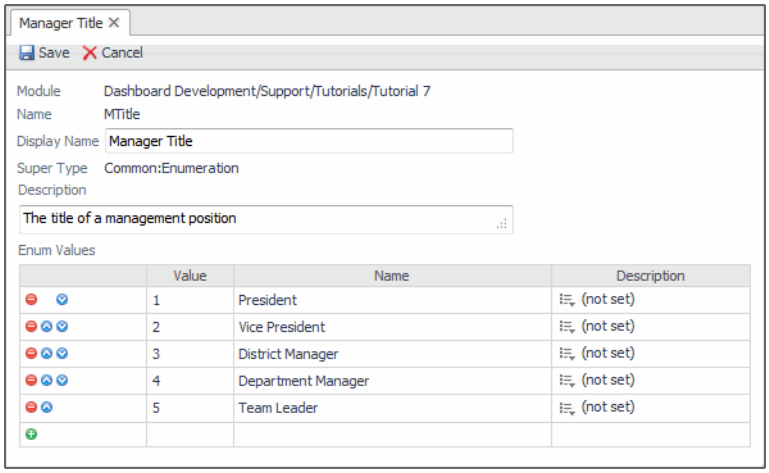
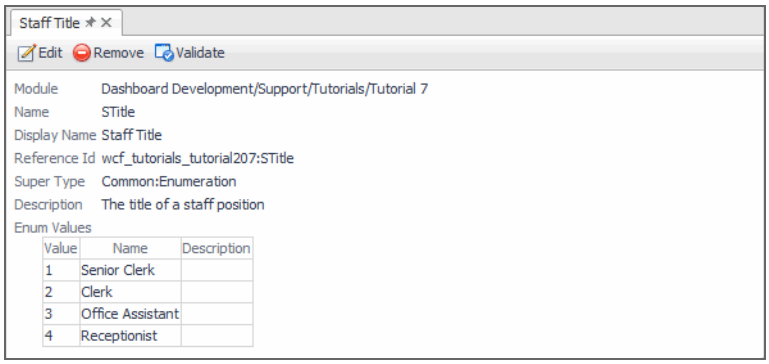


Figure 4. Staff Title type



3 Save the type definitions.

Defining the Manager and Staff Types

To create the Manager and Staff types:

- Still in the types editor, define two more types: `Manager` and `Staff` with the settings displayed below, and save them as usual.

Figure 5. Manager type

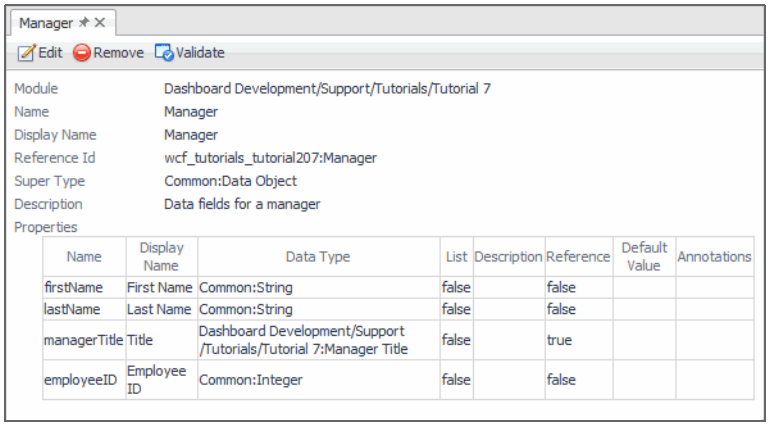
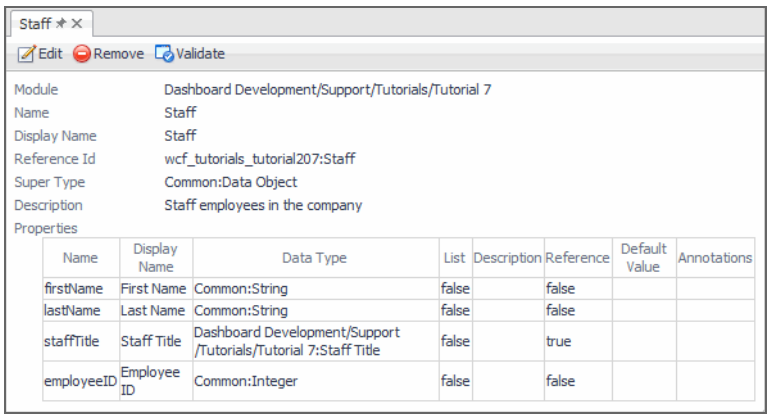


Figure 6. Staff type

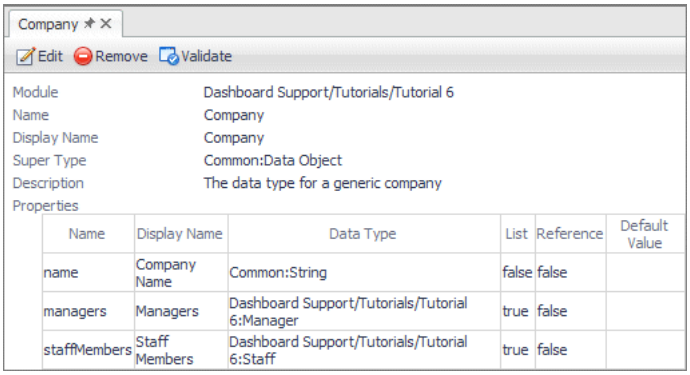


Defining the Company Type

To create a Company type consisting of a Name, list of Managers and list of Staff Members:

- Still in the types editor, define the `Company` type with the settings displayed below, and save it.

Figure 7. Company type



Defining the Functions for Adding and Removing a Manager

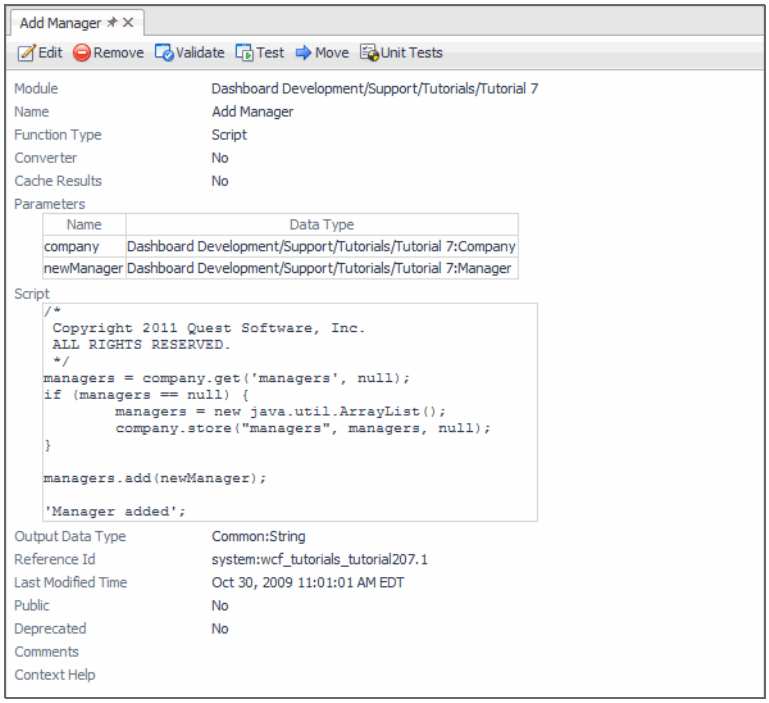
To create the Add Manager function:

1 In the Module Contents pane, click the drop-down list, select **Functions**, and click Add .

The **Add Function** dialog box appears.

2 Create the `Add Manager` function using the information provided below. Define two parameters, *company* and *newManager*, as illustrated below, to pass in values required by the function.

Figure 8. Add Manager function



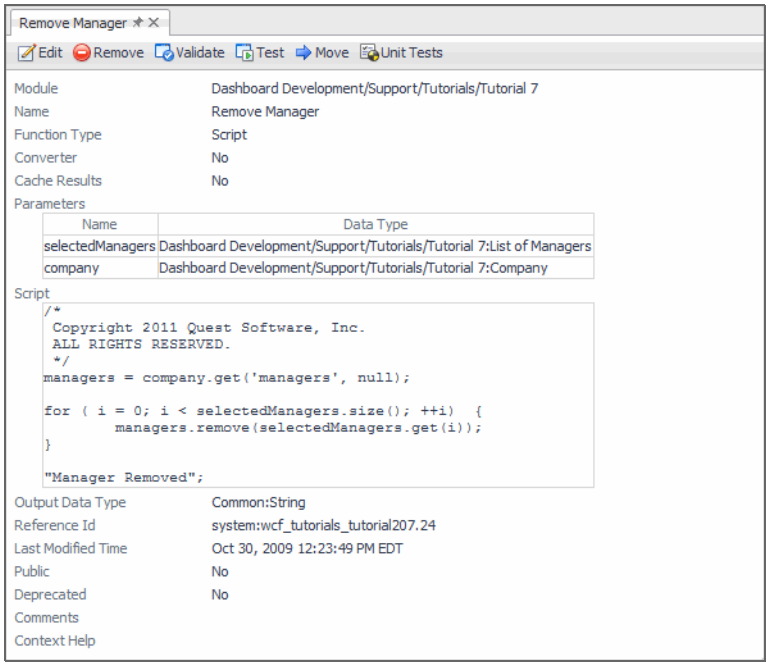
To create the Remove Manager function:

1 In the Module Contents pane, click the drop-down list, select **Functions**, and click Add .

The **Add Function** dialog box appears.

2 Create the `Remove Manager` function using the information provided below.

Figure 9. Remove Manager function



Defining the Labels

To create labels for the Bean Input fields:

1. In the Module Contents pane, click the drop-down list, select **Views**, and click Add .

   The **Add Views** dialog box appears.

2. Create four labels: **Label: First Name**, **Label: Last Name**, **Label: Title**, and **Label: Employee's ID**.

3. When setting their `Label` property on the **Configuration** tab, choose **Set to Binding > Localized String** from the menu, and set the component labels as follows:

   - **Label: First Name**: `First Name:`
   - **Label: Last Name**: `Last Name:`
   - **Label: Title**: `Title:`
   - **Label: Employee's ID**: `Employee's ID:`

4. For each label, set their `Horizontal Alignment` property to `Right`.

Defining the Bean Input Components

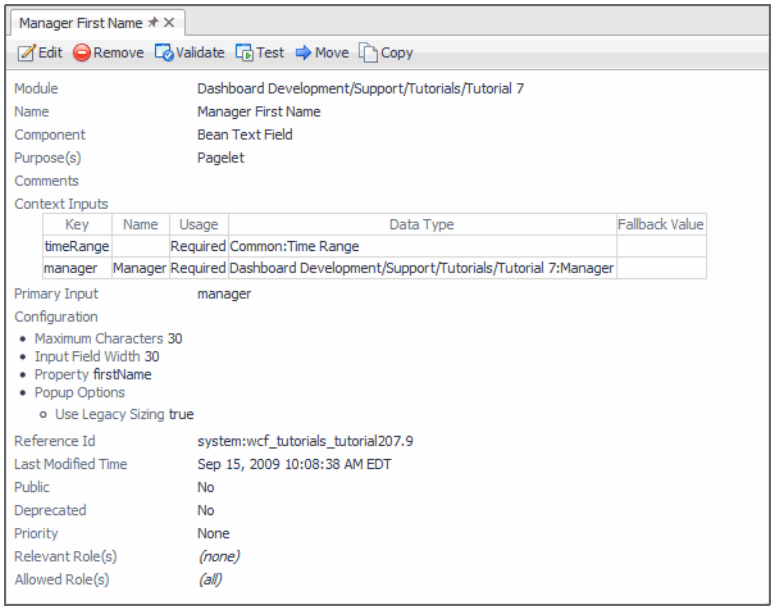
To create the Bean Input components:

1. Start by creating the **Manager First Name** Bean Text Field view. Still in the view editor, click **Add**.

2. In the **Add View** dialog box that appears, on the **Create a Blank View** tab, choose **Bean Inputs > Bean Text Field**.

3. Create the Bean Text Field view using the information provided below.

Figure 10. Manager First Name Bean Text Field



TIP: The `Path To Value` property is easily set because the only context passed to these components is a Manager type.

4 Create two more Bean Input components: the **Manager's Title** Bean Drop-Down List and **Manager's ID Number** Bean Number Input. Use the information provided below to populate the components' values.

Figure 11. Manager's Title Bean Drop-Down List

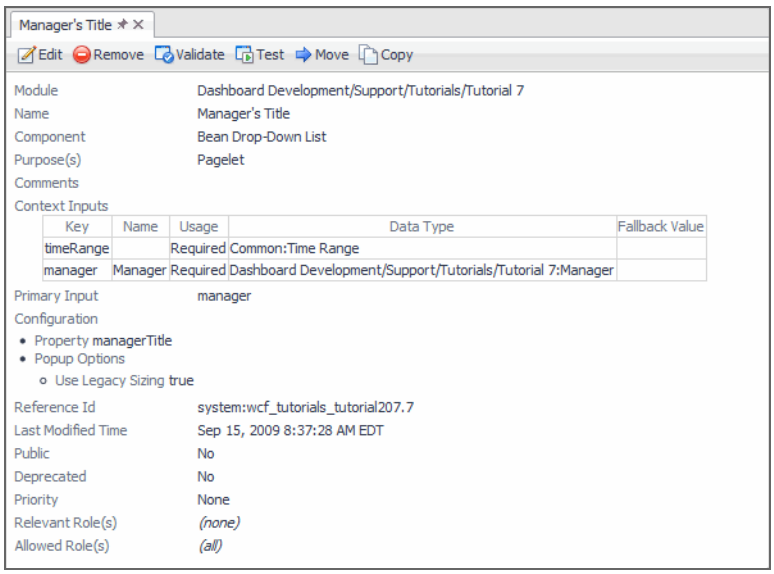
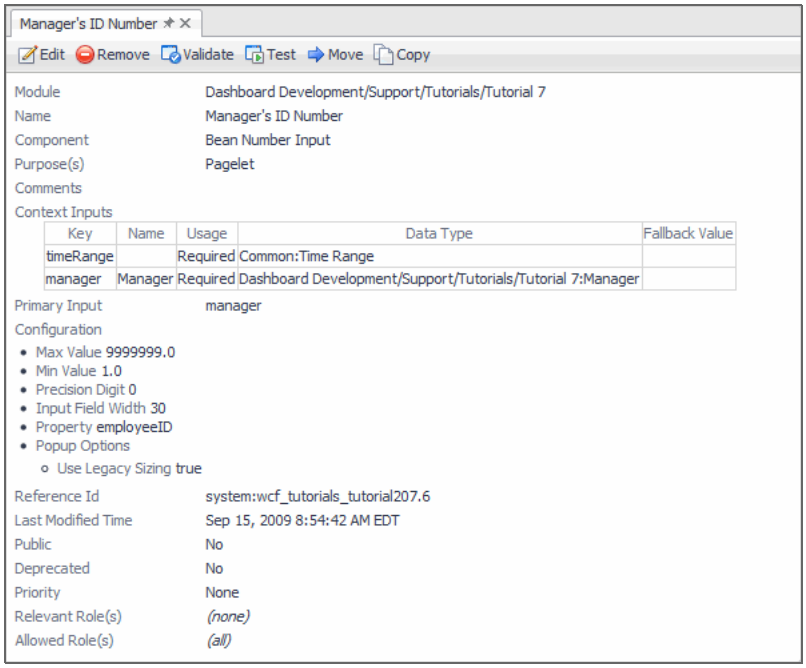


Figure 12. Manager's ID Number Bean Number Input

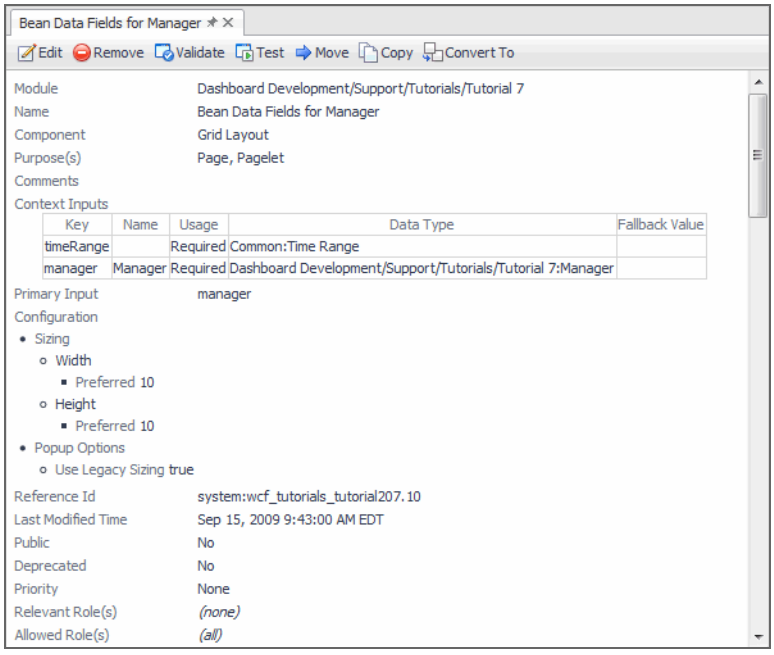


Defining the Grid Layout

To create the Grid Layout for the data collection fields:

- Still in the view editor, create a Grid Layout view using the following values.

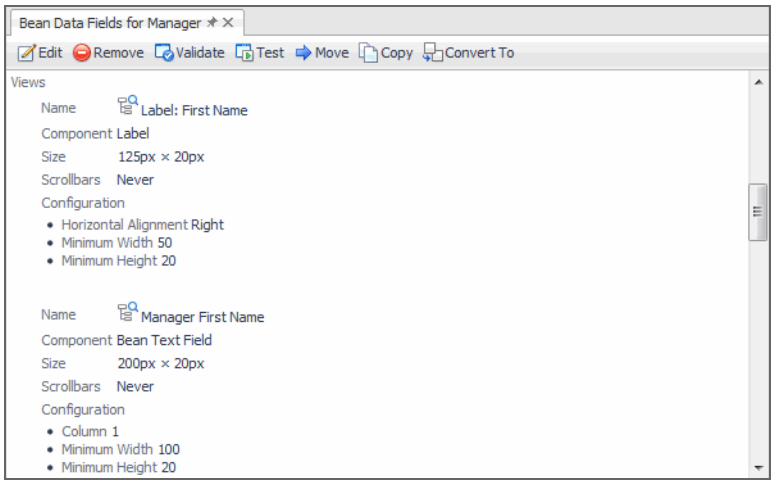
Figure 13. Grid Layout View



IMPORTANT: The manager key is the only required context input.

This tutorial demonstrates the reduction in the number of context-required entries by using Bean Input components. Without them, context keys are required for all the fields in the *Company* structure. If this were a real application, with perhaps dozens of fields, the saving in overhead would be much greater.

Figure 14. First Name Label



The settings for Last Name are similar to those of First Name, but these values are placed in row 1.

Figure 15. Title Label

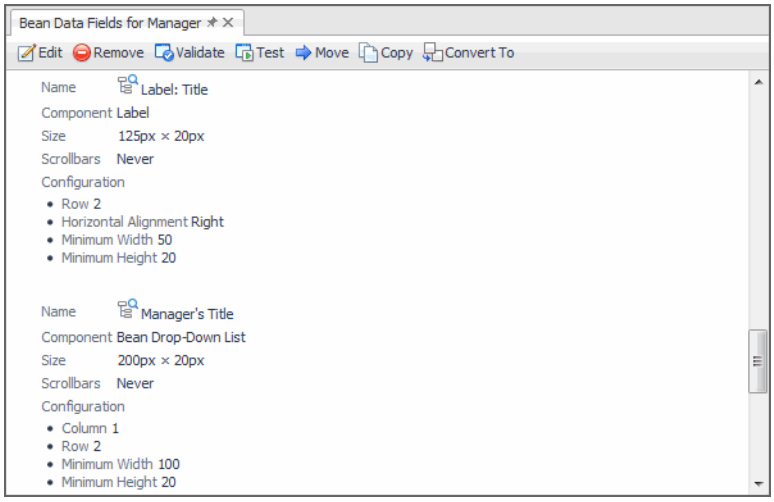
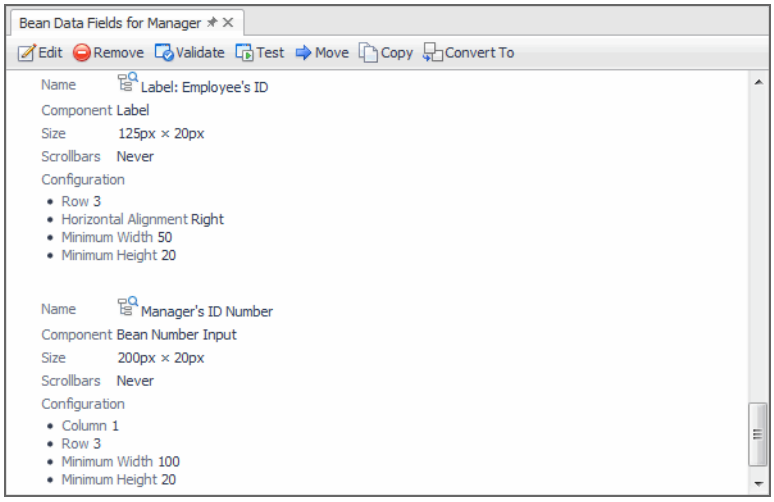


Figure 16. Employee's ID Label

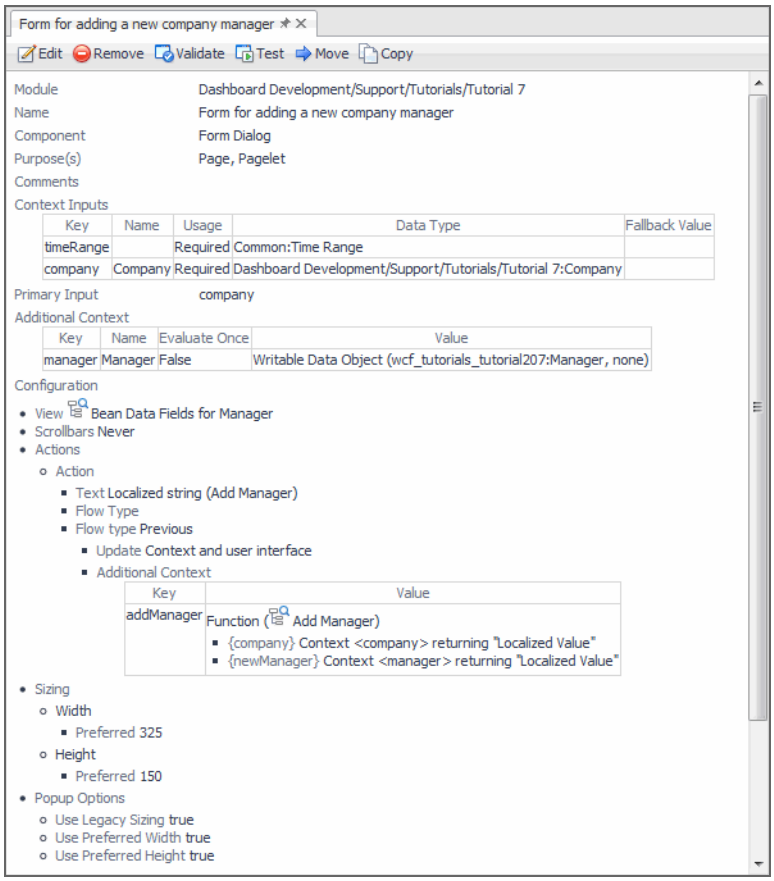


Defining the Dialog

To create the Form for adding a new company manager:

1 Still in the view editor, create a Form Dialog view using the following values.

Figure 17. Form Dialog settings



The storage type for the `manager` object is `none`, which means that each time the Form Dialog view is rendered, a new `manager` object is created, but this object is not saved. It is only when the `company` object is updated and saved that the data gathered here is written into that object, which contains a list of `manager` objects. See the *Web Component Guide* for information about the available storage mechanism choices.

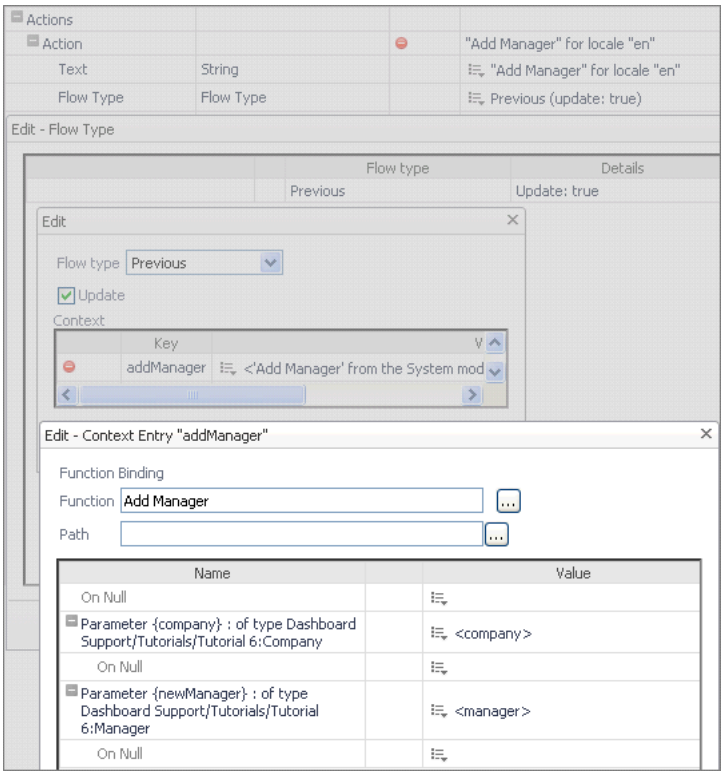
2 Create an Action as shown below.

Notice that the Action is `Add Manager`, accessed by choosing **Invoke Function**. A key (`addManager`) must be supplied, but in this tutorial it is not used anywhere but in the **Flow type** editor.

The required parameters for `Add Manager` (`company` and `newManager`) are associated with the context keys that supply the required values.

The **Flow type** is set to **Previous**, and the **Update** option is selected. When the popup is disposed, the table's data is updated to show the new information.

Figure 18. Flow type settings

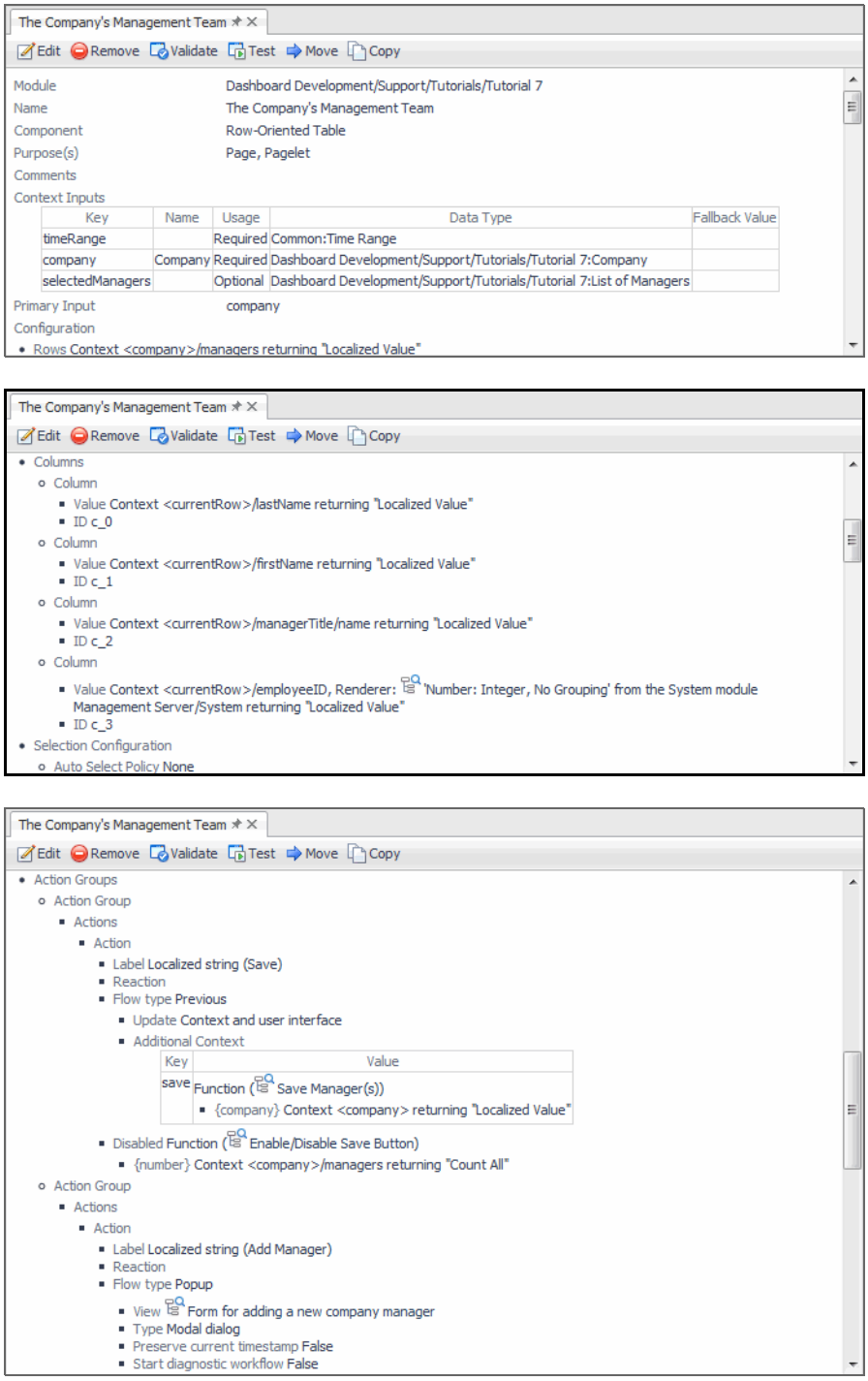


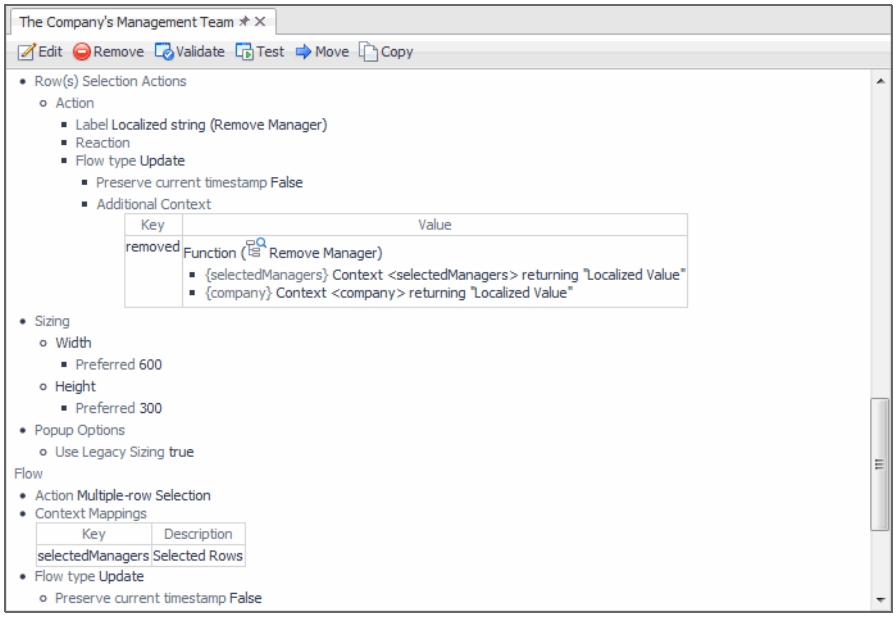
Defining the Table of Managers

To create The Company's Management Team table:

1 Still in the view editor, create a Row Oriented Table using the following values.

Figure 19. Row-Oriented Table settings

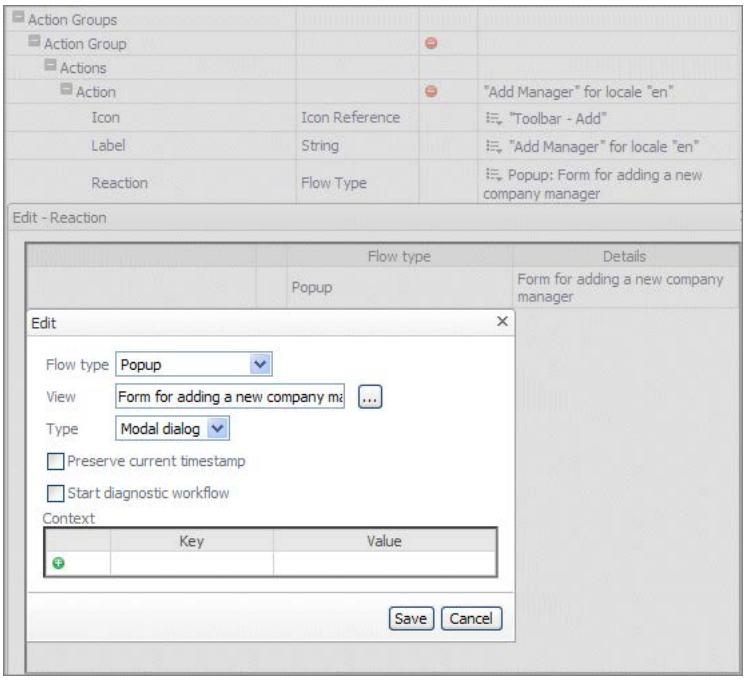




2 Create an Action as shown below.

> **IMPORTANT:** Three nested dialogs are necessary to complete this step. In the third, the Available Views dialog (not shown), you choose the Form for adding a new company manager.

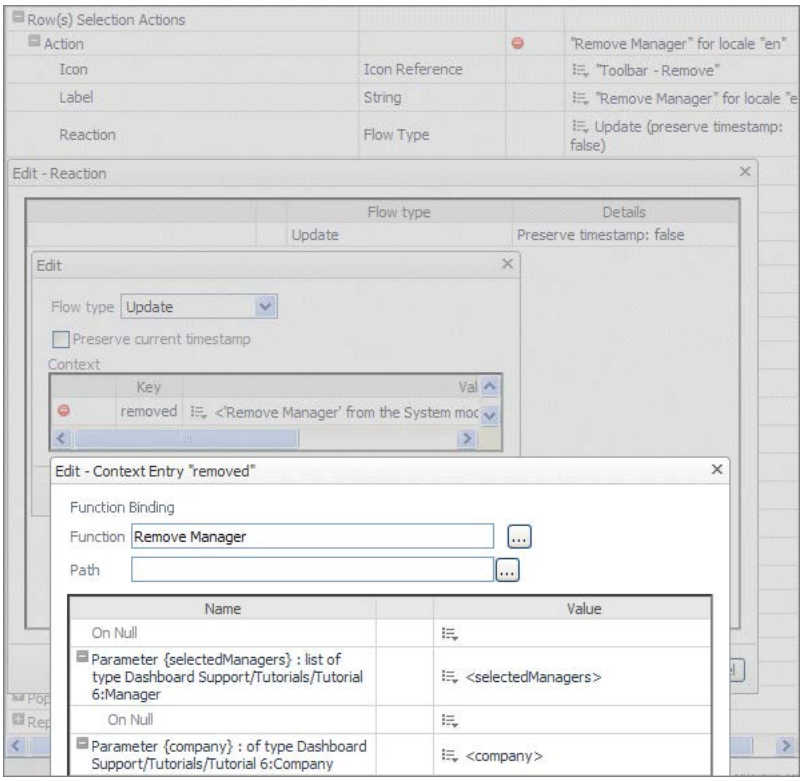
Figure 20. Editing flow actions



3 Create the Row(s) Selection Action as shown below.

> **IMPORTANT:** Five nested dialogs are necessary to complete this step. In the second (Edit) dialog, you choose Invoke Function, and in the third (not shown), you choose Remove Manager from the available list of functions. Back in the Edit - Context Entry “removed” dialog, you choose selectedManager and company as input keys in the Edit - Parameter dialogs (also not shown).

Figure 21. Editing flow actions



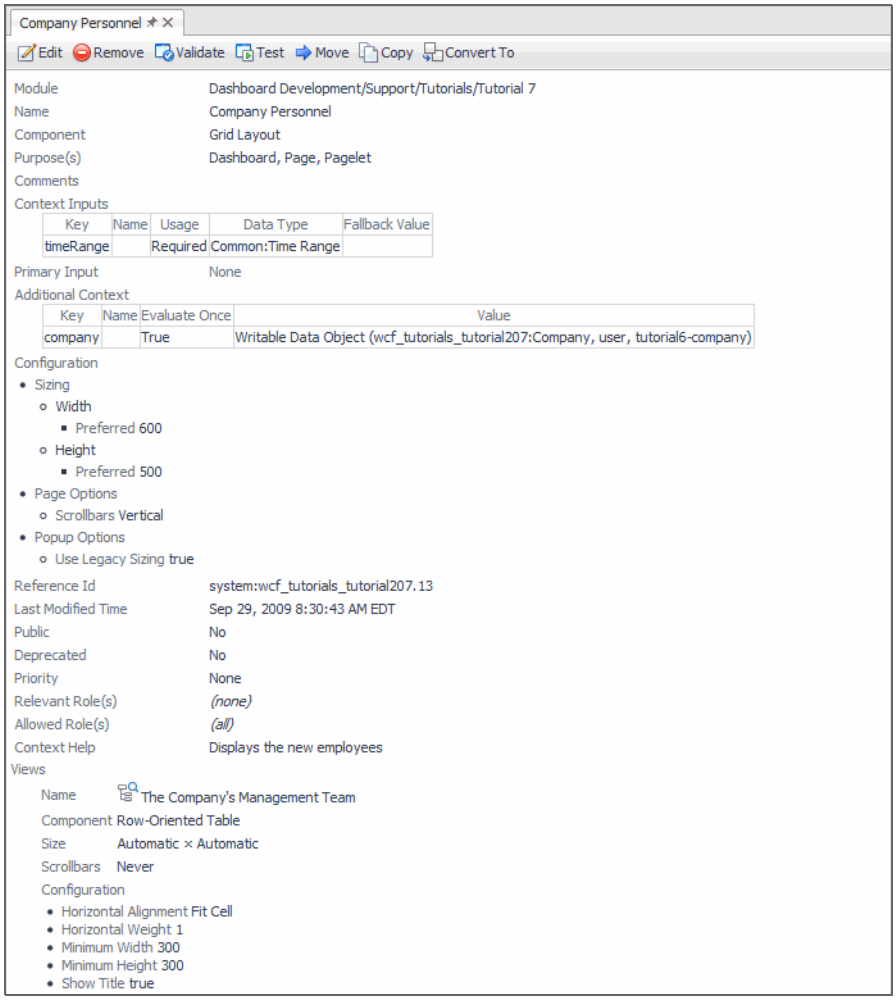
Defining the Top-Level Dashboard

In most designs, the table appears in a grid that holds the table along with other related views. We place the table in such a grid to complete the tutorial. Creating a dashboard is not covered in previous tutorials, so all that needs explaining here is the context entry. A structure is needed to hold the list of managers. By creating an instance of a company object in the dashboard, we also create a manager array that is initially null. Nonetheless, we can get a reference to it and create an empty list of managers the first time the `Add Manager` function is called. After that, `manager` entries can populate the list.

To create a Company Personnel Grid Layout to hold the table:

1 In the view editor, create a Grid Layout using the following values.

Figure 22. Grid Layout Component settings



Note that the storage type for the `company` object is `user`, which means each different logon name has a different `company` object that is kept until the server is restarted. See the *Web Component Guide* for information about the available storage mechanism choices.

Summary

Bean Input components

This tutorial introduces the usage of Bean Input components as a way of collecting data. Notice that very few context keys were declared, which makes it a much more efficient mechanism for collecting and storing data, especially when large amounts of data inputs are required.

Dialog component

If you set a flow on a component in a form, it is triggered independently. This ability, combined with the new smart update functionality, removes any HTML-induced restrictions on how a particular page and its submissions need to be structured.

The proper way to take advantage of this feature is to not use the existing submit action on the form, but to use the Form Dialog component. Additionally, the Wizard component also supports this dual mode of update.

The Form Dialog component allows you to specify multiple actions and draws them in a standard way. If the form is a popup, it automatically renders a cancel button that provides the same functionality as clicking the Close button in the title bar to close the window.

The input elements of the form are then added to whatever layout you need, and added to the Form Dialog itself.

Additional activities

A better design is to decouple saving the information from the act of adding or removing each manager. The functions can be changed so the person doing the data entry adds and removes rows as before, but this time only the table updates. The changes are not saved, and if the user logs out, the data is lost. This design permits a review of all the changes. If it turns out that the user wants to cancel the edits, the user logs out and the changes are lost, but the original data remains.

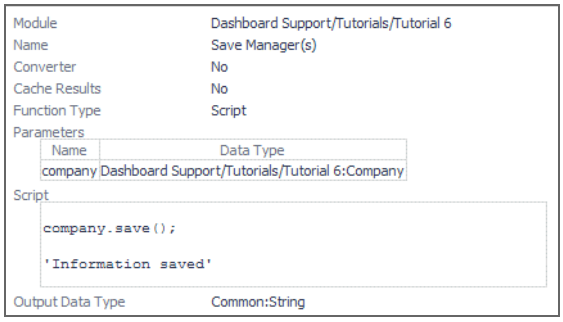
A separate Save button is used to commit the edits to the table.

Add a Save button to the table

Change the way that managers are added and removed. The tutorial is presented in such a way that a manager is added to the database immediately after disposing of the dialog component. Similarly, managers are deleted by choosing Remove Manager. To allow for the changes to be reviewed, change the functions so that updating the manager list is a separate action triggered by a button on the table toolbar. The table updates as managers are added and deleted, but the changes are not be committed until the Save button is clicked.

This is a particularly easy change to make. All you need to do is move the Groovy statement `company.save()` to a new function called *Save Manager(s)* and invoke it from a new table toolbar button called *Save*.

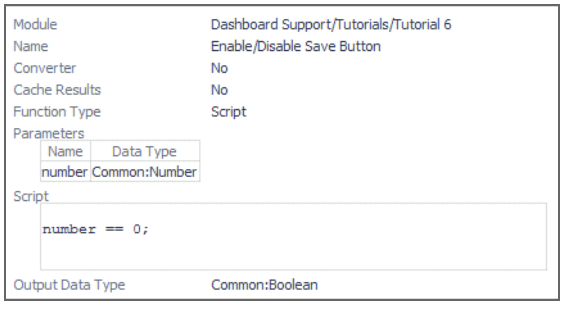
Figure 23. Save Manager function settings



Control the state of the Save button

The Save button should only be enabled if the table is not empty. Thus, the Save button should be disabled before any managers are added, enabled when there is at least one manager, and disabled again if all rows are removed. Create the function as shown in the figure:

Figure 24. Enable/Disable Save Button function settings



In the `Save` action, invoke the **Enable/Disable Save Button** with the key `number`, and use an input key of `company` and path `managers`. Set the return type to **Count All**.

NOTE: The final version of this tutorial is available in Dashboard Support > Tutorials > Tutorial 7 > Company Personnel if you give the module an Allowed Role of Dashboard Designer by editing the module in the Configuration > Definitions > Definitions pane.

9

Tutorial 8: Renderers

As their name implies, renderers are used to give the final appearance to the object that they display. All data types in the framework have default renderers associated with them, but there are times when you need to choose a different representation. In this case, choose either an existing renderer other than the default, or create a specialized version of one of the following renderer types:

- Date
- Error*
- Host Name
- Icon
- List
- Metric Bar
- Metric Baseline
- Null Image*
- Null String*
- Number
- Number Bar
- Number Percent
- Number Unit
- Progress
- Severity
- Severity Counts
- Sparkline
- SQL
- Style
- Text
- Timeline
- Time Range

In most cases, the names of these renderers are sufficient to indicate their function, but you can read the full description, and see a sample image, in the *Web Component Reference*.

NOTE: Most renderers have an input value that uses a context setting whose key is value (if it is set) as its input value. Otherwise, it uses its own default value. For details, see Input value.Renderers marked with an asterisk (*) do not have configurable input values.

By now you are familiar with using the Definitions editor to configure components by setting their properties. This tutorial dispenses with step-by-step procedures and leaves the configuration up to you. You are encouraged to build the necessary components as you proceed through the tutorial.

- Text Renderer

- Date Renderer
- Number Bar Renderer and Number Unit Renderer

Objectives of Tutorial 8

These are the main topics in this tutorial:

- Show how to use a Text renderer to modify a text style. For more information, see Text Renderer on page 118.
- Show the different styles possible with a Date renderer. For more information, see Date Renderer on page 119.
- Show examples of Number and Number Unit renderers. For more information, see Number Bar Renderer and Number Unit Renderer on page 120.

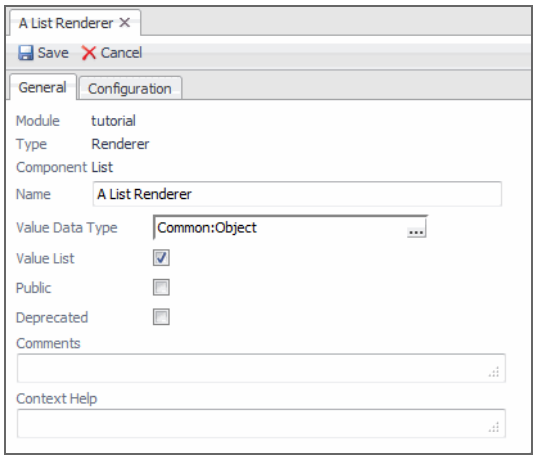
Input value

Many renderer types support the concept of an Input Value. This is the value to be rendered and is made available in the context under the key `value`.

If you edit a renderer that supports Input Values, you can specify on the **General** tab the **Value Data Type**, and through **Value List** whether the value is a list. You can then refer to the Input Value through a Context Selection binding by setting the key to `value`. You can also drill down on the context entry in order to access sub-properties of that type.

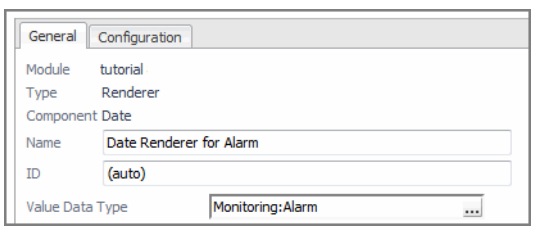
The following figure shows an example of a List renderer.

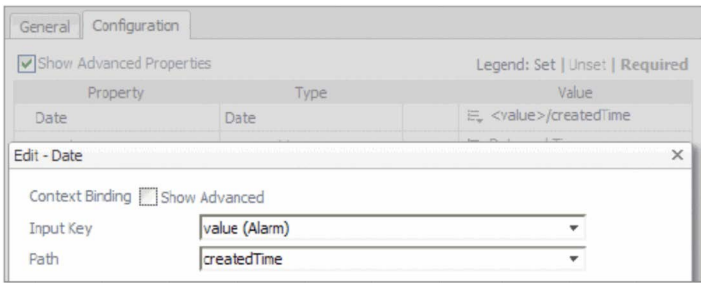
Figure 1. List renderer settings



Another example, this time for a Date Renderer, shows that if you set the **Value Data Type** to `Monitoring:Alarm`, then you can edit the `Date` property to set it to a Context Selection Binding with an **Input Key** of `value`, and **Path** of `createdTime` in order to render the *Created Time* property of an alarm as a date.

Figure 2. Date renderer settings





The default values of **Value Data Type** depend on the renderer component. The default value of **Value List** is `False`, except for the List and Severity counts renderers.

Text Renderer

Each component that presents text has a default renderer associated with it. The renderer specifies the properties, such as font size and font style, that the component uses to display the text.

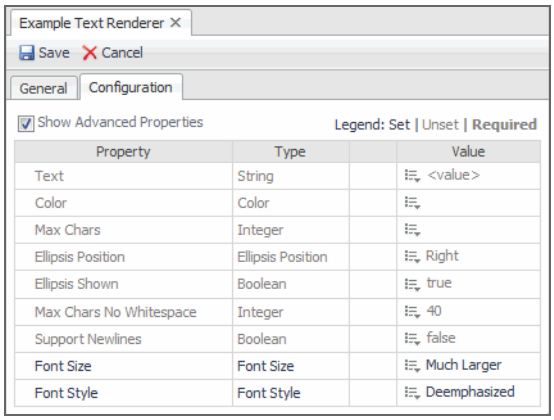
If you need to display text in a component in a manner that is different from the default, you can create your own renderer.

To configure a new renderer:

1 Create a new Text renderer.

2 Change the renderer properties, as applicable.

Figure 3. Text renderer settings



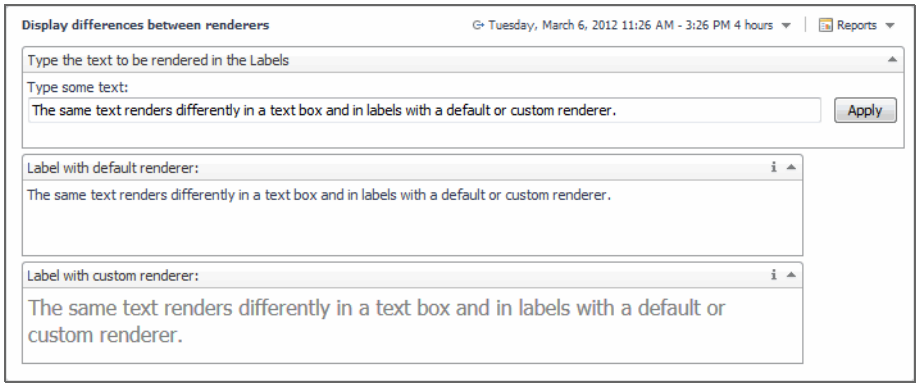
IMPORTANT: It is recommended to associate colors by using themes (for details about color themes, see the Web Component Guide). Do not define the renderer color by setting the Color property in the Configuration tab, as this may result in colors clashing with the background or the text disappearing entirely, if a user changes the theme.

3 Edit your component, for example, a Label. When configuring the `Label` property that displays the label text, associate it with your custom text renderer.

The following figure shows the differences among three components:

- Text Field
- Label component and its default renderer
- Label component and its custom renderer with the **Font Size** set to `Much Larger` and **Font Style** to `Deemphasized`

Figure 4. Displaying newly created renderers



Working with the Text Renderer

The properties and their allowed values are described in the Renderer Components pages of the *Web Component Reference*. If you look up *Font Size*, you see that it is an enumerated type with four allowed values. You can set the property to one of these as a static value using **Set to Enum Value**, or you can use bindings in **Set to Binding**. If you use **Set to Binding** you must take care that the binding does supply one of the allowed values.

You can limit the number of characters that the renderer displays and you can insert an ellipsis (…) to indicate that the string is truncated.

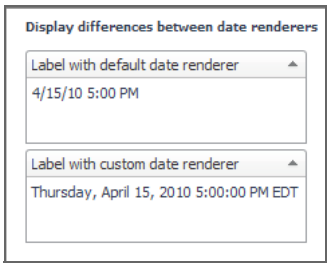
You can use the **Max Chars No Whitespace** property to insert a space in extra long strings, which may help you to format the output in certain cases.

Date Renderer

The Date renderer component encodes both date and time.

The figure shows how a date appears when the default renderer is used, and how it appears when both `Display Length` and `Time Display Length` are set to `Full`.

Figure 5. Displaying date renderers



The renderer can be used to display only the date or only the time.

The `Custom Pattern` property, based on Java™ date formats, can be set, or the `Fixed Format` property, based on ISO 860. These properties are described in the *Web Component Reference* page describing the Date renderer.

Number Bar Renderer and Number Unit Renderer

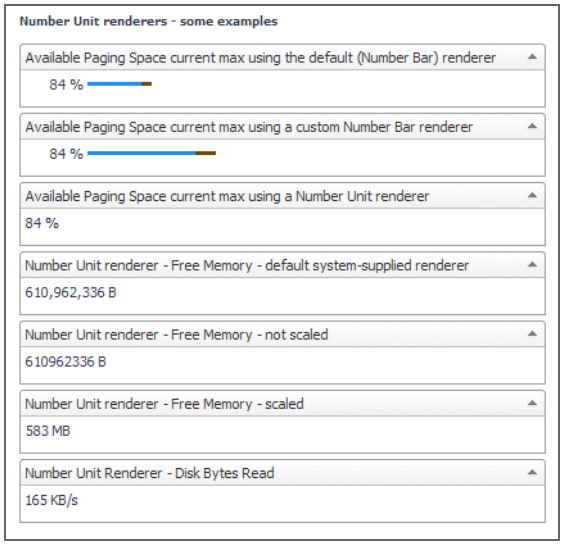
The Number Bar renderer can be used to show a graphical representation of the number in relation to its maximum possible value.

The Number Unit renderer contains numerous properties. You are encouraged to read the “Number User Guide” page in the “Renderer Components” subsection of the *Web Component Reference*. You can control the number of decimal places in the display, the number’s rounding properties, how digits are grouped in big numbers, and provide a scaling factor.

There are default Number Unit renderers in Foglight® for the numerical quantities with units associated with them. You can change how a number and its unit appear by creating a new Number Unit renderer.

The following figure shows some examples of default renderers, and what can be accomplished by creating your own renderer.

Figure 6. Default renderers



Description of each label:

- **Available Paging Space current max using the default (Number Bar) renderer**: No renderer is assigned. The system-supplied default renderer appends a percent sign and a bar as instructed by the system.
- **Available Paging Space current max using a custom Number Bar renderer**: Just to show how you can change the default behavior, the custom Number Bar renderer in this example stretches the length of the bar that shows the percentage of used paging space.
- **Available Paging Space current max using a Number Unit renderer**: This time a Number Unit renderer is used instead of a Number Bar renderer. The number is expressed as a percentage, but it is positioned closer to the left margin.
- **Number Unit renderer, Free Memory, Default system-supplied renderer**: The system supplies a version of the Number Unit renderer that displays the free JVM memory. The default renderer uses grouping and displays the unit.
- **Number Unit renderer - Free Memory - not scaled**: A custom version of the Number Unit renderer displays a number and its associated unit, but grouping is not enabled.
- **Number Unit renderer - Free Memory - scaled**: The renderer and the associated unit are scaled.
- **Number Unit renderer - Disk Bytes Read**: The way in which the renderer retrieve its value is what is different in this example. Normally the input value to a Number Unit renderer is a number, but there are

times when that number is not directly available from the context. See Another example of working with an input value.

Configuring the Labels

Renderers are used for display, and in this tutorial the components display the numbers are labels. The objects supplying the numbers are:

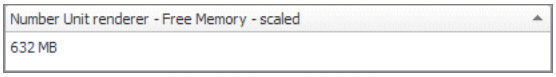
- **Number Bar renderer**: A host object whose data type is `Host`, chosen because it supplies the `Available Paging Space` metric, whose unit is percent
- **Number Unit renderer**: An FSM Server of type `CatalystServer`, chosen because it supplies the `JVM Free Memory` metric, whose unit is number of bytes

Since there are three labels of each type, one query is constructed to supply the host and another to supply the FSM Server.

NOTE: The queries should be constructed to supply top-level objects like hosts and FSM Servers, even though the query itself could drill into the desired property. The parent property supplies the unit, so configuring the proper path in the context setting ensures that the unit is properly assigned.

To configure the labels:

1. Construct a grid to hold the labels. Declare two internal context inputs, `fsmServer` and `host`, whose fallback values are queries that return the first objects in their respective lists.
2. Construct the renderers. For example, the **Number Unit renderer - Free Memory - scaled label** uses these property values:
   - **Show Unit**: `true`
   - **Auto Scale Unit**: `true`
   - **Number**: Context `<value>`
   - **Use Grouping**: `true`
   - **Significant Digits**: `3`
3. Construct the labels. For example, the **Number Unit renderer - Free Memory - scaled** label uses these settings:
   - Context input key: `fsmServer`
   - **Label** property: `first item of Context <fsmServer>/jvm/freeMemory/current/max`
   - Renderer: `Number Unit Renderer - Free Memory - scaled - query returning "Localized Value"`
4. Add the label to the grid so that you can view the renderer's output.



Another example of working with an input value

This example illustrates how a renderer's input value can be used to select the necessary data type. The value displayed in Number Unit Renderer - Disk Bytes Read was found as follows:

1. Consider that this is the only available query:
   - **Query name**: `Return a Metric - Disk Bytes Read`
   - **Query definition**: `Select objects of type Metric from the Monitoring data source (of type foglight-5) from Root of (foglight-5:default)/HostModel/hosts/storage/diskBytesRead`

The query returns a list object of the `Monitoring:Metric` type, but it is the `current/max` property that should be displayed. The renderer can be configured to drill down to the required number.

The first item of the query, which is a `Metric`, is available through the context key `metric`.

2 Construct a Number Unit renderer, changing the default data source type and data type:

- **Name**: `Number Unit Renderer - Disk Bytes Read`
- **Value Data Type**: `Monitoring:Metric`
- **Configuration**:
  Show Unit: `true`
  Number: `Context <value>/current/max returning "Localized Value"`

To get to the renderer's `Number` property, pass in a `Metric` value and then drill down to the required number, which is the metric's `current/max`.

3 Construct the label by setting its **Label** property as follows:

- **Label**: Context `<metric>`
- **Renderer**: `Number Unit Renderer - Disk Bytes Read returning "Localized Value"`

Add the label to the grid and test the result.

10

Tutorial 9: Adding Questions and Answers

The Question Viewer component allows you to display a list of questions, organized into categories, and the answer to a selected question.

This tutorial leads you through the process of creating domain-specific questions, defining their answers, and viewing these using the Question Viewer.

- Working with Associations and Question Viewers
- Building Associations for Your Question Viewer
- Populating the Question Viewer with Questions and Answers
- Building a Mini-Viewer
- Summary

Objectives of Tutorial 9

The objectives of this tutorial are to introduce the Question Viewer, Question Mini-Viewer, and the related Association components. You will learn about the Question Viewer and Question Mini-Viewer components, as well as Question, Category, and Domain Associations.

The tasks you complete in this tutorial are:

- Learn about Associations.
- Create Domain and Category Associations.
- Define questions and answers about specific parts of your monitored system, where the answer points to a view that answers the question, and each question and answer pair belongs to a domain, and is assigned to at least one category.
- Create a Question Viewer component and associate it with the domain in order to retrieve all of questions that are associated with that domain. The resulting view shows the categories, questions, and answers in a three-column view.
- Create a Questions Mini-Viewer and associate it with a category in order to retrieve all of questions that are associated with that category. The resulting view shows one question at a time, allowing you go back and forth between the questions and display the answers.

Working with Associations and Question Viewers

Associations are WCF entities. They organize references to other entities and enable specific functionality in view components. Associations are an important part of creating questions and the Question Viewer component.

There are three types of associations you work with when creating a Question Viewer:

- Domain

- Question
- Category

These associations provide functionality such as high-level question grouping, links between questions and answers, and navigation between questions, as well as enabling you to specify the content of your questions.

Before configuring a Question Viewer component, you need to create at least one Domain association, one Category association, and one Question association.

You should become familiar with Domain, Question, and Category associations before proceeding with the tutorial. The sections that follow help you understand these types of associations and how they are used by the Question Viewer component. For complete information about Domain, Category, and Question Associations, and the Question Viewer, see the *Web Component Reference*.

About Domain Associations

The first kind of association you create as part of this tutorial is a Domain association.

A Domain is a conceptual grouping that you create and name. It should be on a scale that corresponds with a broad grouping of related objects or an environment type that you are interested in monitoring. For example, in this tutorial, you create a Domain called *Host Domain*, which represents a host object provided as context input. You can also have Domain entities called *Databases* or *Application Servers*.

Domain associations provide high-level grouping for questions. You specify a Domain association for each question. This Domain association broadly filters the set of all questions to those appropriate for your Question Viewer. For example, the Question Viewer you create in this tutorial is focused on questions about a host object.

About Question Associations

You use Question associations to specify the questions and answers that you want to display in one or more Question Viewers.

You specify the question itself as plain text and the answer as a referenced view.

You can also specify a report view that is used if the Question is added to a report. This is an optional property of the Question Association. To learn more about it, refer to Additional activities on page 135.

About Category Associations

You assign one or more categories to your questions. Categories create sub-groups and help users navigate a set of questions.

You use Category Associations to group related questions within the context of a domain. For example, in this tutorial you create two categories, *CPU* and *Memory*, for questions related to a host's CPU and memory utilization, which allow you to organize questions within your broader *Host Domain*.

About Question Viewers

Question Viewers display a list of questions, sorted by category, and the answer for a selected question. A Question Viewer must point to a domain which enables the viewer to retrieve the questions associated with the domain. The Question Viewer also must any have context input that satisfies all of the views that contain the answers

Building Associations for Your Question Viewer

Follow the instructions below to create the three types of Associations your *Host Question Viewer* will use: Domain, Question, and Category. In this example, the Domain Association, *Host*, has two categories of questions, *CPU* and *Memory*, with one question-answer pair in each, as illustrated below. The answers are linked to basic chart views defined in the *Host* module, *Host Memory Utilization* and *CPU Utilization*, that take a *Host* object as input. For that reason, the Question Association components, *CPU Utilization* and *Memory Utilization*, need to take a *Host* object as context input.

In addition to the component name, the Domain Association, *Host*, and the two Category Associations, *CPU* and *Memory*, require configuring two properties: *Display Name* and *Canonical Name*. You will use these properties in the Question Associations to point to the domain. The Question Associations, *CPU Utilization* and *Memory Utilization* are slightly more complex in that they need include references to the Domain and Category Associations, and the views that provide answers to the questions. They also need to include context input (a single *Host* object in this example) for the views they reference.

Component structure

The following diagram illustrates the hierarchy between the associations and views and the overall structure that you are about to create.

Table 1. Domain Association *Host*

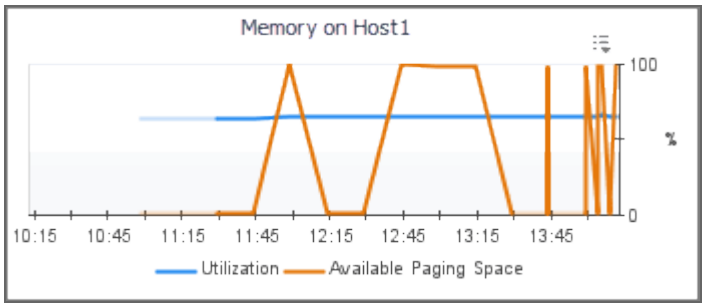
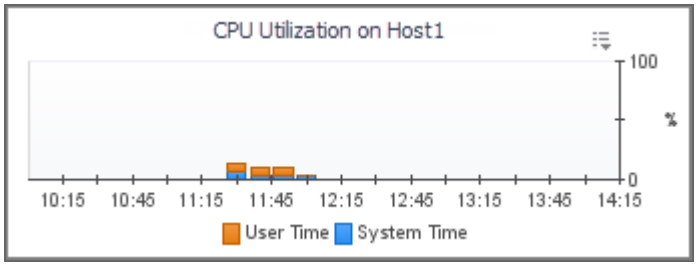
Category Association *CPU*	
Question Association *CPU Consumption*	
Question	
	What is the CPU utilization on this host?
Answer	
	
Category Association *Memory*	
Question Association *Memory Utilization*	
Question	
	What are the memory utilization and available paging space on this host?

Table 1. Domain Association *Host*

	Answer
	

Creating a Domain

In this section, you create a Domain Association, *Host*. In addition to the component name, the *Host* domain requires a display name and a canonical name. The display name is the domain name that appears in the browser interface. The canonical name is used for grouping and matching. You will use the canonical name later in this tutorial to associate this domain with the Question Viewer and the mini-viewer.

To create a domain:

1 Start by creating a new Domain Association.

   a On the navigation panel, under **Dashboards**, choose **Configuration** > **Definitions**.

   b In the Module List Pane, select the module or sub-module in which you want to store the components you create while using this tutorial.

   c In the Module Contents pane, click **Views** and select **Associations** from the list that appears.

   d Click .

   e In the **Add Association** dialog box that appears, select **Domain** and click **OK**.

   A **New Association** tab appears in the Definition/Edit pane with the **General** tab open.

   f Specify the name for this association. In the **Name** box, type `Host`.

2 Specify the display name of the Domain Association. This is the displayed domain name.

   a Open the **Configuration** tab.

   b In the row containing the **Display Name** property, click the **Value** column and select **Set to String** from the list that appears.

   c In the **Edit - Display Name** dialog box that appears, in the **String Value** box, type `Host`.

   d Click **Set**.

3 Specify the canonical name of the Domain Association. This name is used for grouping and matching. Later in this tutorial, we will use this name to associate the Question Viewer with this domain.

   a On the **Configuration** tab, in the row containing the **Canonical Name** property, in the **Value** column, click Edit , and select **Set to String** from the list that appears.

   b In the **Edit - Display Name** dialog box that appears, in the **String Value** box, type `myHostDomain`.

   c Click **Set**.

4 On the **New Association** tab, in the top-left corner, click **Save**.

Creating Categories

In this section, you create a two Category Associations, *CPU* and M*emory*. Like the Domain Association you just created, these categories also require display names and canonical names.

To create categories:

1 Start by creating a new Category Association.

   a In the Module Contents pane, ensure that **Associations** is selected in the top-left, and click .

   b In the **Add Association** dialog box that appears, select **Category**, and click **OK**.

   A **New Association** tab appears in the Definition/Edit pane with the **General** tab open.

   c Specify the name for this association. In the **Name** box, type `CPU`.

2 Specify the display name of the Category Association. This is the displayed category name.

   a Open the **Configuration** tab.

   b In the row containing the **Display Name** property, in the **Value** column, click Edit , and select **Set to String** from the list that appears.

   c In the **Edit - Display Name** dialog box that appears, in the **String Value** box, type `CPU`.

   d Click **Set**.

3 Specify the canonical name of the Category Association. This name is used for grouping and matching.

   a On the **Configuration** tab, in the row containing the **Canonical Name** property, in the **Value** column, click Edit , and select **Set to String** from the list that appears.

   b In the **Edit - Display Name** dialog box that appears, in the **String Value** box, type `myHostDomain`.

   c Click **Set**.

4 On the **New Association** tab, in the top-left corner, click **Save**.

5 Create another Category Association, *Memory*.

6 Follow Step 1 through Step 4 to create the Memory Association with the following property values:

   - **Name**: `Memory`
   - **Display Name**: `Memory`
   - **Canonical Name**: `hostMemory`

Creating Questions

In this section, you create a two Question Associations, *CPU Utilization* and M*emory Utilization*. Questions are slightly different in terms of configuration. First, you associate each question with the *Host* domain and the appropriate category (*CPU Utilization* with *CPU*, and *Memory Utilization* with *Memory*). Next, point each question to the view that provides the answer. Finally, configure the context input. Because each of these views take a *Host* object as input, the question associations need to have a *Host* object as input to pass that information to the views.

To create questions:

1 Start by creating a new Question Association.

   a In the Module Contents pane, ensure that **Associations** is selected in the top-left, and click .

   b In the **Add Association** dialog box that appears, select **Question** and click **OK**.

   A **New Association** tab appears in the Definition/Edit pane with the **General** tab open.

   c Specify the name for this association. In the **Name** box, type `CPU Utilization`.

2 Add a `Host` context input key to the Question Association.

In this Question Association we will use the *CPU Utilization* view as the answer. Because this view displays the total percentage of time a host's CPU executes system code and user programs, it takes a `Host` object as input. For that reason, we need to pass a `Host` object to this association as a context input key.

a Open the **Context** tab.

b Click to add an entry to the **Inputs** table.

c In the newly added row, in the **Key** column, type `Host`.

d Click the **Data Type** column and click the selector button (...) that appears.

e In the **Select Type** dialog box that appears, select the **Host** row and click **Save**.

The **Select Type** dialog box closes and the **Data Type** column refreshes, showing **Monitoring:Host** in the newly added row.

3 Specify the question for this Question Association.

a Open the **Configuration** tab.

b In the row containing the **Question** property, in the **Value** column, click Edit, and select **Set to String** from the list that appears.

c In the **Edit - Question** dialog box that appears, in the **String value** box, type `What is the CPU utilization on this host?`

d Click **Set**.

4 Specify the canonical name of the domain that you want to associate with this the Question Association. In this example, we want to associate this question with the **Host** domain, whose canonical name is `myHostDomain`.

a On the **Configuration** tab, in the row containing the **Domain Canonical Name** property, in the **Value** column, click Edit.

b In the **Edit - Domain Canonical Name** dialog box that appears, navigate to the module containing the Domain Association definition, and select its `Host` node.

c Click **Set**.

5 Specify the answer to this question. In this case, we want to associate this question to the **CPU Utilization** view available in the **Hosts > Host** module.

a In the row containing the **Answer View** property, in the **Value** column, click Edit.

b In the **Edit - Answer View** dialog box that appears, in the Name column, navigate to the **CPU Utilization** view by choosing **Hosts > Host > [chart] CPU Utilization**.

c Click **Set**.

6 Associate this question with a category. In this example, we want to add this question to the **CPU** group of questions.

TIP: A question can appear and be associated with one or more categories.

a Expand the **Categories** property node and click to add a new category.

b In the row containing the **Association Reference** property that appears, in the **Value** column, click Edit,.

c In the **Edit - Category** dialog box that appears, in the **Name** column, navigate to the module containing **CPU** Category Association, and select it.

d Click **Set**.

7 On the **New Association** tab, in the top-left corner, click **Save**.

8 Create another Question Association, *Memory Utilization*.

9 Follow Step 1 through Step 7 to create the Question Association with the following property values:

- **Name**: `Memory Utilization`
- **Context Input**: `Host`
- **Question**: `What are the memory utilization and available paging space on this host?`
- **Domain Canonical Name**: `myHostDomain`
- **Answer View**: `Hosts/Host/[chart] Host Memory Utilization`
- **Categories**: `Memory`

Populating the Question Viewer with Questions and Answers

Now that you have the associations required to populate the Question Viewer, you can start building the Question Viewer component that we will name *Host Question Viewer*. First, ensure that the viewer is a Page or Pagelet. In a typical scenario, a Question Viewer appears on a dashboard and displays questions and answers about the objects that appear on that dashboard. For that reason, there is no need to make the Question Viewer a dashboard.

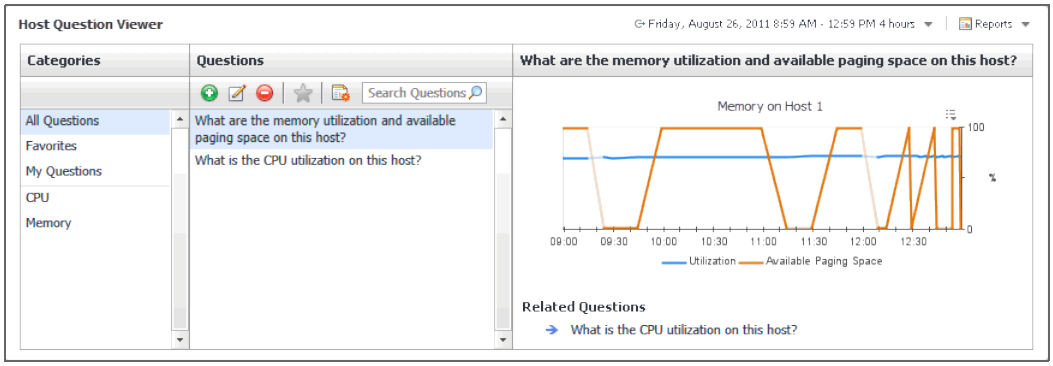
Next, ensure that the *Host Question Viewer* has context input that satisfies the views that contain the answers. In this example, we use the *Host Memory Utilization* and *CPU Utilization* views, that both take a *Host* object as input. For that reason, the *Host Question Viewer* needs to have a *Host* object as a context input as well.

Finally, configure the viewer to point to the *Host* domain. This causes the Question Viewer to retrieve the questions associated with the domain.

In a typical environment, the Question Viewer uses a service to retrieve the related domain associations. The service is the viewer's context input key to the Question Viewer which populates the list of domains using real data. While placing the Question Viewer on a dashboard and creating a service to populate the list of domains is possible, it requires using the Service Builder and creating a service category which is beyond the scope of this tutorial. For that reason, the final product of this tutorial is not a dashboard (without input), but a simple Page/Pagelet containing this Question Viewer which points to the domain simply by using its canonical name, *myHostDomain*, we specified earlier in this tutorial.

The Question Viewer you build in this section contains three columns: **Categories**, **Questions**, and **Question**. You can customize the collection of questions by adding more questions or marking questions as favorites for future reference. You can also search the entire collection of questions for a specific question using a search box.

Figure 1. Host Question Viewer



- **Categories**: This column displays three default categories, and the categories you specified as the Category Associations. The default categories are **All Questions**, **Favorites**, and **My Questions**. The categories you specified in this tutorial, **CPU** and **Memory**, appear just below the default categories.

- The **All Questions** category is selected by default and contains all of the questions defined in the Question Viewer, including the questions you define in the viewer component along with any questions you add at run-time using the Add button in the **Questions** column.
- The **Favorites** category contains the questions you mark using the Favorites button in the **Questions** column.
- **My Questions** column contains the questions you add at run-time using the Add button in the **Questions** column.

Note When **My Definitions** is not the working module, and you add a question to **My Questions**, the question does not appear in this category. It can be found in **All Questions**.

- **Questions**: This column contains the questions that belong to the category selected in the **Categories** column.
- **Answer**: This column contains the view that represents the answer to the question selected in the **Questions** column. This is also known as the answer area. The answer to a question is a portlet view.

To build the Question Viewer:

1 Start by creating a new Question Viewer component.

a In the Module Contents pane, ensure that **Views** is selected in the top-left, and click .

b In the **Add View** dialog box that appears, on the **Create a Blank View** tab, in the navigation tree, choose **Others > Questions Viewer**, and click **OK**.

A **New View** tab appears in the Definition/Edit pane with the **General** tab open.

c Specify the name for this association. In the **Name** box, type `Host Question Viewer`.

2 Add a `Host` context input key to the Question Viewer.

The Question Viewer references questions that have the *CPU Utilization* and *Host Memory Utilization* views as answers. These views display host metrics and for that reason need a *Host* object as input. Therefore, the viewer needs to include the same input to satisfy the referenced views.

a Open the **Context** tab.

b Click to add an entry to the **Inputs** table.

c In the newly added row, in the **Key** column, type `Host`.

d Click the **Data Type** column and click the selector button (...) that appears.

e In the **Select Type** dialog box that appears, select the **Host** row and click **Save**.

The **Select Type** dialog box closes and the Data Type column refreshes, showing **Monitoring:Host** in the newly added row.

3 Associate the Question Viewer with the *Host* domain.

The Question Viewer needs to point to the *Host* domain to populate it with the questions that are associated with the domain. For that purpose, we use a WCF function, *List of Domain References from Canonical IDs,* that retrieves a list of domain references using their canonical names. Earlier in this tutorial, we specified the canonical name of the *Host* domain.

a Open the **Configuration** tab.

b In the row containing the **Domains** property, in the **Value** column, Edit .

c In the list that appears, select **Function**.

d In the **Edit - Domains** dialog box that appears, in the **Function** box, click ...

e In the **Available Functions** dialog box that appears, in the **Name** column, in the navigation tree, choose **Dashboard Development > Support > Common > List of Domain References from Canonical IDs**, and click **Save**.

The **Available Functions** dialog box closes, and the **Edit - Domains** dialog box refreshes, showing the **List of Domain References from Canonical IDs** in the **Function** box.

f Specify the **IDs** parameter.

In a typical environment, the Question Viewer appears on a dashboard and displays information about a selected object. The viewer uses this function to retrieve domain associations from a service that is a context input key to the Question Viewer. The domain associations are list properties (*domainAssociationIDs*) of the service. The list populates the *IDs* parameter of the *List of Domain References from Canonical IDs* function.

In this tutorial, the Question Viewer does not appear on a dashboard. We defined it as a simple Page/Pagelet component. While placing the Question Viewer on a dashboard and creating a service to populate the list of domains is possible, it requires using the Service Builder and creating a service category which is beyond the scope of this tutorial. Instead, for the purpose of this exercise, we populate the **Domains** property using a simple list with one text string, *myHostDomain*, which is the canonical name of the *Host* domain.

For complete information about services in *Foglight*, see the *Foglight User Guide*.

In the row containing the **Parameter {ids}** entry, in the **Value** column, click Edit .

g In the list that appears, select **Set to Binding > List**.

h In the **Edit - Parameter** dialog box that appears, leave all the three check boxes cleared, and click **Set**.

i In the **Edit - Domains** dialog box, Click under the parameters row.

j In the **Parameter row,** click .

k In the list that appears, select Localized String.
The Localized String option may not appear because this option is not available for the My Definitions module. Select String Template instead.

l In the Edit -Parameter dialog box that appears, type myHostDomain in the Localized String box, the canonical name of the Host domain we specified earlier in this tutorial.

m Click Set to close the Edit -Parameter dialog box, and then click Set to close the Edit -Domains dialog box.

4 On the **New View** tab, in the top-left corner, click **Save**.

5 Test the Question Viewer.

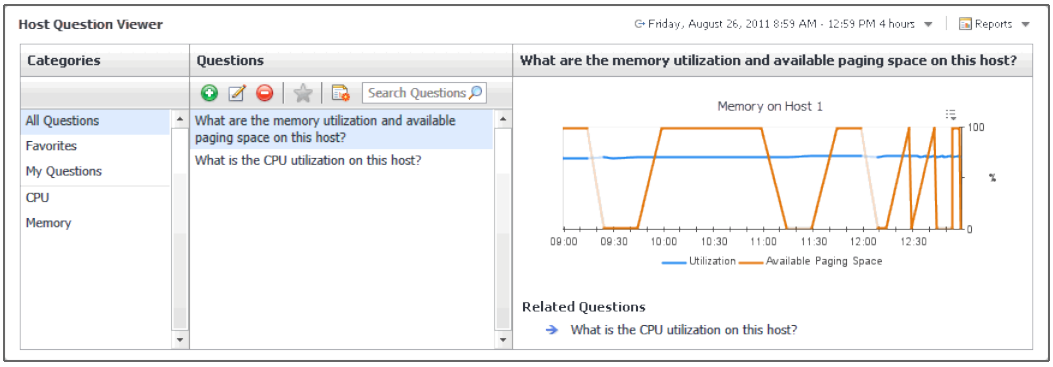
a In the Definition Editor, navigate to the *Host Question Viewer* definitions, and click **Test**.

Specify the host that you want to scope on with the Question Viewer. In the **Input Values** dialog box that appears, in the **Host** box, click Edit , and select a host object from the list that appears.

b Click **Results**.

A new tab appears (or a new window, depending on your browser type), showing the Question Viewer in the display area with the **All Questions** category selected in the **Categories** column.

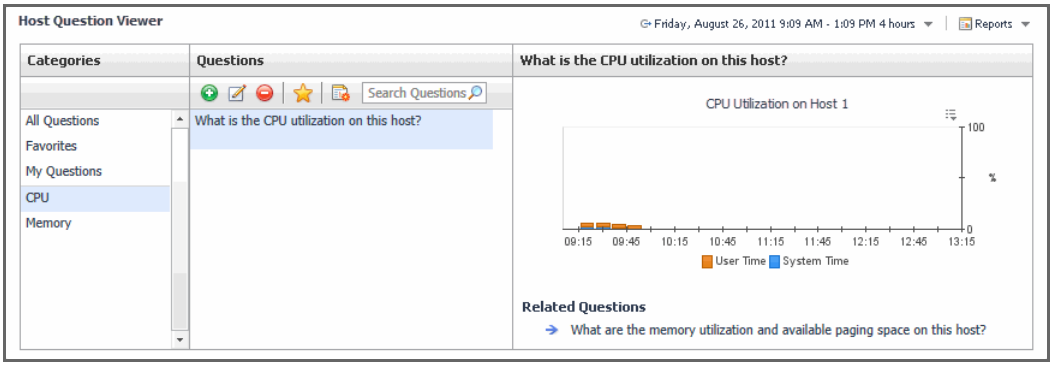
Figure 2. All Questions



c In the Question Viewer, in the **Categories** column, click **CPU**.

The **Questions** column refreshes, showing only the question that belongs to this category, **What is the CPU Utilization on this host?** The question appears selected in the **Questions** column, and the *CPU Utilization* view appears on the right, showing the CPU utilization for the selected host.

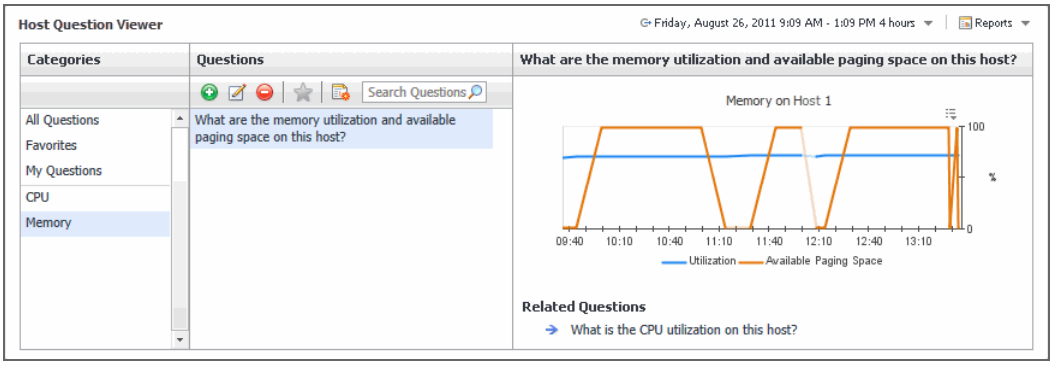
Figure 3. Selecting a question category



d In the Question Viewer, in the **Categories** column, click **Memory**.

The **Questions** column refreshes, showing only the question that belongs to this category, **What are the memory utilization and available paging space on this host?** The question appears selected in the **Questions** column, and the *Host Memory Utilization* appears on the right, showing the memory utilization and available paging space for the selected host.

Figure 4. Selecting another question category



Building a Mini-Viewer

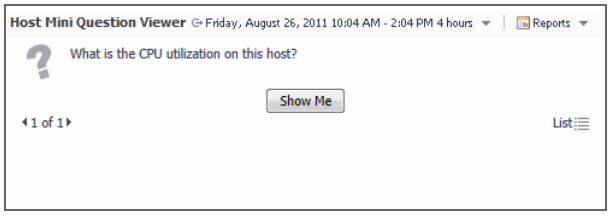
Question Mini-Viewers are very similar to the Question Viewers in that they also display information about a selected object using Question and Category Associations. This is a smaller view component that can appear on a dashboard to display information about a selected object.

Unlike the Question Viewer that references a domain, the initial view of the Question Mini-Viewers uses a Category Association as a filter, and can only display the questions associated with that category. In this example, we use the existing associations: the *CPU* category and its *CPU Utilization* question to display in the mini-viewer. It also allows you to see all of the questions associated with the domain in a secondary view.

Like in the previous example, we first need to ensure that the viewer is a Page or Pagelet, and that it takes a *Host* object as input (required by the *CPU Utilization* view). Next, configure the mini-viewer to point to the *Host* domain using its canonical name, *myHostDomain*.

The mini-viewer you build in this section contains the *CPU Utilization* question. You select the question by navigating the list using the left and right facing arrows in the bottom-left. In this example, the category associated with the mini-viewer contains only one question. As an exercise, consider adding more questions to the *CPU* category to see how the navigation works.

Figure 5. Question mini-viewer



To build a mini-viewer:

1 Start by creating a new Question Mini-Viewer component.

   a In the Module Contents pane, ensure that **Views** is selected in the top-left, and click .

   b In the **Add View** dialog box that appears, on the **Create a Blank View** tab, in the navigation tree, choose **Others > Questions Mini-Viewer**, and click **OK**.

   A **New View** tab appears in the Definition/Edit pane with the **General** tab open.

   c Specify the name for this association. In the **Name** box, type `Host Mini Question Viewer`.

2 Add a `Host` context input key to the Question Mini-Viewer.

   Again, we are providing a *Host* object as input to satisfy the *CPU Utilization* view.

   a Open the **Context** tab.

   b Click to add an entry to the **Inputs** table.

   c In the newly added row, in the **Key** column, type `Host`.

   d Click the **Data Type** column and click the selector button (...) that appears.

   e In the **Select Type** dialog box that appears, select the **Host** row and click **Save**.

   The **Select Type** dialog box closes and the Data Type column refreshes, showing **Monitoring:Host** in the newly added row.

3 Associate the mini-viewer with the *CPU* category. In this example, we want to add this question to the **CPU** group of questions.

   a Open the **Configuration** tab.

   b In the row containing the **Category** property, in the **Value** column, click Edit ,.

c In the **Edit - Category** dialog box that appears, in the **Name** column, navigate to the module containing **CPU** Category Association, and select it.

d Click **Set**.

4 Associate the mini-viewer with the *Host* domain.

Like in the previous procedure, we need to point to the *Host* domain using the function *List of Domain References from Canonical IDs* and providing *Host* domain's canonical name, *myHostDomain*, as the function parameter in a single-entry list.

a On the **Configuration** tab, in the row containing the **Domains** property, in the **Value** column, Edit .

b In the list that appears, select **Function**.

c In the **Edit - Domains** dialog box that appears, in the **Function** box, click Edit .

d In the **Available Functions** dialog box that appears, in the **Name** column, in the navigation tree, choose **Dashboard Development > Support > Common > List of Domain References from Canonical IDs**, and click **Save**.

The **Available Functions** dialog box closes, and the **Edit - Domains** dialog box refreshes, showing the **List of Domain References from Canonical IDs** in the **Function** box.

e In the row containing the **Parameter {ids}** entry, in the **Value** column, click Edit .

f In the list that appears, select **Localized String**.

g In the **Edit - Parameter** dialog box that appears, in the **Localized String** box, type `myHostDomain`. This is the canonical name of the *Host* domain we specified earlier in this tutorial.

h Click **Set** to close the **Edit - Parameter** dialog box.

i In the **Edit - Domains** dialog box, click **Set** to close it.

5 On the **New View** tab, in the top-left corner, click **Save**.

6 Test the mini-viewer.

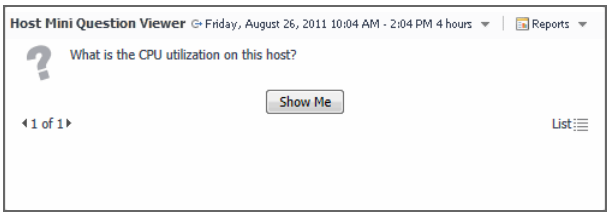
j On the **Host** Mini tab containing the mini-viewer definition, click **Test**.

Specify the host that you want to scope on with the Question Viewer. In the **Input Values** dialog box that appears, in the **Host** box, click Edit and select a host object from the list that appears.

k Click **Results**.

A new tab appears (or a new window, depending on your browser type), showing the mini-viewer in the display area, and the *CPU Utilization* question, **What is the CPU Utilization on this host?**

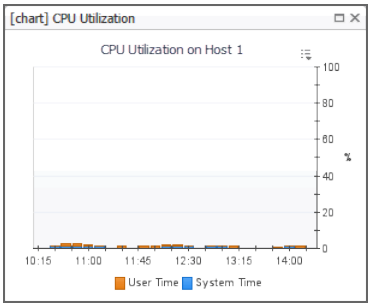
Figure 6. Selecting a question in the mini-viewer



l In the mini-viewer, click **Show Me**.

A dialog box appears, showing the *CPU Utilization* view.

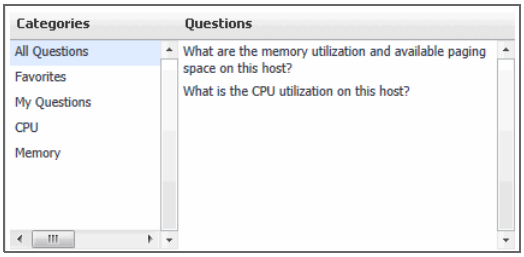
Figure 7. Viewing an answer in the mini-viewer



m Close the dialog box.

n In the Question Mini-Viewer, click **List**.

Figure 8. Listing question categories in the mini-viewer



A secondary view appears. It shows all of the categories associated with the domain that is referenced by this view.

o Click through the different categories, selecting the individual questions and observing the answer views.

Summary

In this tutorial you have been shown how to create Domain, Category, and Question Associations, and to use them to populate a Question Viewer.

Additional activities

After completing this tutorial, consider performing the following activities:

- Adding more questions to the CPU category and reviewing them in the mini-viewer
- Adding report views
- Creating custom answer views

11

Tutorial 10: Sending Messages to Other Users

Messages inform other users of any activities that may affect them. A message includes the message text and message details. It can also have a severity level to indicate the overall impact of the activity that is about to take place.

- Creating Message Associations
- Creating Script Functions
- Creating Label Views
- Testing Messages
- Summary

Objectives of Tutorial 10

The objectives of this tutorial are to introduce Message Associations and Script Functions. Use Message Associations in your application to set the desired message text and details. Script Functions can reference this information to send messages when invoked.

The tasks you complete in this tutorial are:

- Create Message Associations with sample message text and details.
- Create Script Functions that send messages by referencing the Message Associations.
- Create Label views that call the Script Functions in the their flow.
- Test messages by placing the Label views on a dashboard and invoking them from there.

Creating Message Associations

In this section we will create two message associations: *Warning Association* and *Error Association*. These associations will have the Warning and Error severity levels, and the related message text and details.

To create Message Associations:

1 Start by creating a new Message Association.

   a On the navigation panel, under **Dashboards**, choose **Configuration** > **Definitions**.

   b In the Module List Pane, select the module or sub-module in which you want to store the components you create while using this tutorial.

   c In the Module Contents pane, click **Views** and select **Associations** from the list that appears.

   d Click .

   e In the **Add Association** dialog box that appears, select **Message** and click **OK**.

   A **New Association** tab appears in the Definition/Edit pane with the **General** tab open.

f Specify the name for this association. In the **Name** box, type `Warning Association`.

2 Specify the severity level of the Message Association.

a Open the **Configuration** tab.

b In the row containing the **Level** property, in the **Value** column, click Edit . Select **Data** from the list that appears.

c In the **Edit - Name** dialog box that appears, in the **Name** box, navigate to **Dashboard Development > Support > Common > Message Levels > Warning**.

d Click **Set**.

3 Specify the message text.

a On the **Configuration** tab, in the row containing the **Text** property, in the **Value** column, click Edit . Select **Localized String** from the list that appears.

b In the **Edit - Text** dialog box that appears, in the **Localized string** box, type `Warning`.

c Click **Set**.

4 Specify additional message details.

a In the row containing the **Details** property, in the **Value** column, click Edit . Select **Localized String** from the list that appears.

b In the **Edit - Text** dialog box that appears, in the **Localized string** box, type `This is a warning message`.

c Click **Set**.

5 Save your changes.

6 Follow Step 1 through Step 5 to create another Message Association with the following definitions:

- **Name**: `Error Association`
- **Context Input**: None
- **Level**: `Data (Message Level: Error)`
- **Text**: `"Error" for locale "en"`
- **Details**: `"This is an error message" for locale "en"`

The Message Associations are created. The next step of the tutorial is to create a Script Function that takes these Message Associations as input, and send messages when invoked.

Creating Script Functions

Script Functions execute scripts and return desired results. In this section we will create two Script Functions: *Send Warning Message* and *Send Error Message*. These functions call the *sendMessage()* function and that uses a Message Association as input. This function is defined in the *FunctionHelper* interface available in the WCF API.

For the purpose of this example, we are making a call to the *sendMessage()* function when a label is selected in order to display a message in the browser interface and illustrate the usage of Message Associations. However, in a typical scenario, you will likely use messages in your code, for example, to inform the users that a background process has finished. This can be done by calling the *sendMessage()* function from the code instead of creating a Script Function and invoking it from a view.

For complete information about the *sendMessage()* function and the WCF API, see the *WCF API Reference* in the online help, accessible from the *Getting Started* page under **Developing Foglight Cartridges**.

To create Script Functions that call the sendMessage() function:

1 Start by creating a new Script Function.

d In the Module Contents pane, click **Associations** and select **Functions** from the list that appears.

e Click .

f In the **Add Function** dialog box that appears, on the **Blank Function** tab, select **Script** and click **OK**.

A **New Function** tab appears in the Definition/Edit pane.

g Specify the name for this Script Function. In the **Name** box, type `Send Warning Message`.

2 In the **Script Text** box, type the following text:

```
package system._wcf_tutorials_tutotial2010.scripts;
functionHelper.sendMessage("system:wcf_tutorials_tutotial2010.7", 'site',
   null,'never', 0, null);
```

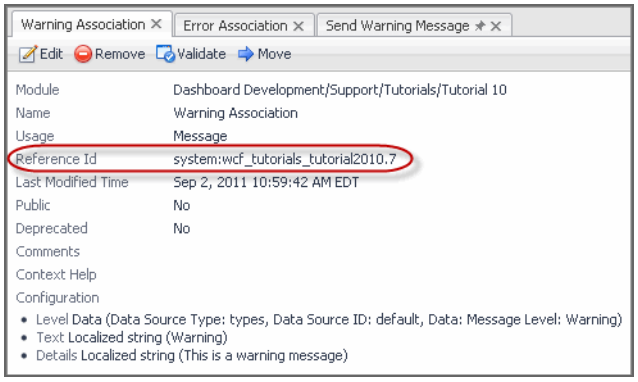
The package name in the first line of code references your working module. If you are working in a different module, there is no need to edit that line.

The second line calls the *sendMessage()* function. The first argument shown in the above example, `"system:wcf_tutorials_tutotial2010.7"`, should contain the reference ID of the Warning Association. Edit that value to reflect the reference ID.

TIP: You can quickly find the reference ID of the Warning Association when you have it open in view mode in the Definition/Edit pane.

For more information about this function, see the *WCF API Reference* in the online help, accessible from the *Getting Started* page under **Developing Foglight Cartridges**.

Figure 1. Warning Association reference ID



3 Observe the **Output Data Type** property.

By default, this Script Function returns an Object. This object will contain the message that will be passed as an input key to the flow of the Label component later in this tutorial. This action results in a new message.

4 Save your changes.

5 Create another Script Function using the same steps, that references the Error Association.

The Script Functions are created. Next, create Label views that invoke these Script Functions in their flow.

Creating Label Views

In this section you will create two Label views: *Warning Label* and *Error Label*. Each of these Labels will invoke the appropriate Script Function in their flow.

To create Label views:

1 Start by creating a new Label view.

   h In the Module Contents pane, click **Functions** and select **Views** from the list that appears.

   i Click .

   j In the **Add View** dialog box that appears, on the **Create a Blank View** tab, choose **Common > Label** and click **OK**.

   A **New View** tab appears in the Definition/Edit pane.

   k Specify the name for this Label. In the **Name** box, type `Send Warning Label`.

2 Specify the label text.

   a Open the **Configuration** tab.

   b In the row containing the **Label** property, in the **Value** column, click Edit , and choose **Set to Value > String**.

   c In the **Edit - Label** dialog box that appears, in the **String name** box, type `Send a Warning Message`.

   d Click **Set**.

3 Enable the triggering of the *Send Warning Message* function when the Label is selected.

   The *Send Warning Message* returns an Object. This object contains the message that will be passed as an input key to the flow. For that reason, we need to add an input key to receive this Object instance.

   a Open the **Flow** tab.

   b Click **Selection** in the list.

   c In the **Edit - Selection** dialog box that appears, clear the **Leave unspecified** check box.

   d Click **Flow type** and select **Update** from the list that appears.

   e In the **Context** table, click to add a new context key.

   f In the **Key** column, type the key name. For example, `message`.

   g In the **Value** column, click Edit , and choose **Set to Binding > Function**.

   h In the **Edit** dialog box that appears, click the selector button (). In the **Available Functions** dialog box that appears, navigate to the *Send Warning Message* function.

   > **TIP:** You will find the function in your current working module.

4 Save your changes.

5 Follow Step 1 through Step 4 to create another Label view with the following definitions:

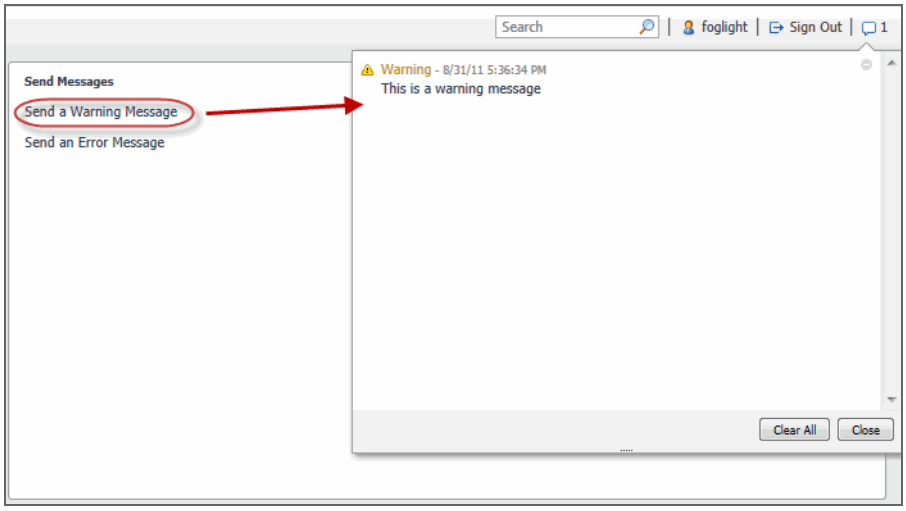
   - **Name**: `Error Label`
   - **Label**: `Send an Error Message`
   - **Update Flow**: Configure it to include a context key pointing to the *Send Error Message* function.

The Label views are created. Next, add these labels to a dashboard and send test messages.

Testing Messages

In this section you will create a simple dashboard, *Send Messages*, and add the Label views to the layout. You will then test the messaging actions by invoking Warning and Error messages and observe the resulting flow.

Figure 2. Testing messages



To test messaging actions:

1 Start by creating a new dashboard.

   i In the Module Contents pane, ensure that **Views** is selected.

   j Click .

   k In the **Add View** dialog box that appears, on the **Create a Blank View** tab, choose **Common > Grid Layout**. Click **OK**.

   A **New View** tab appears in the Definition/Edit pane.

   l Specify the name for this dashboard. In the **Name** box, type `Send Messages`.

2 Add Label views to the dashboard layout.

   a Open the **Layout** tab.

   b Click **Add**.

   c In the **Add View** dialog box that appears, ensure that **Selecting existing view** is selected and click **Next**.

   d Navigate to your working module and select the **Warning Label** from the tree.

   e Accept the default sizing properties and click **Finish**.

   f Follow Step b through Step e to add the *Error Label* to the layout and save your changes.

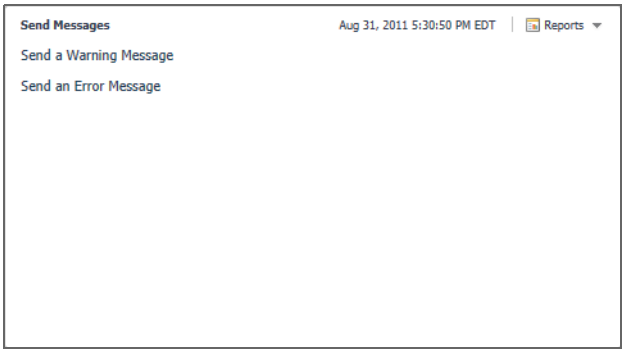
3 Test the messaging action.

   g In the Definition Editor, navigate to the *Send Messages* definitions. Click **Test**.

   The Send Messages dashboard appears in the display area.

   The dashboard contains two links: **Send a Warning Message** and **Send an Error Message**.

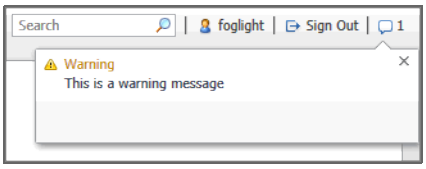
Figure 3. Send Messages dashboard



h Click **Send a Warning Message**.

After a few moments, a Warning message appears in the top-right corner.

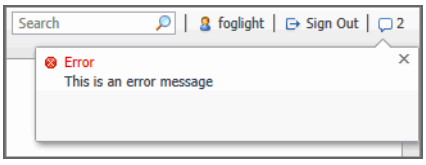
Figure 4. Warning Message



i Click **Send an Error Message**.

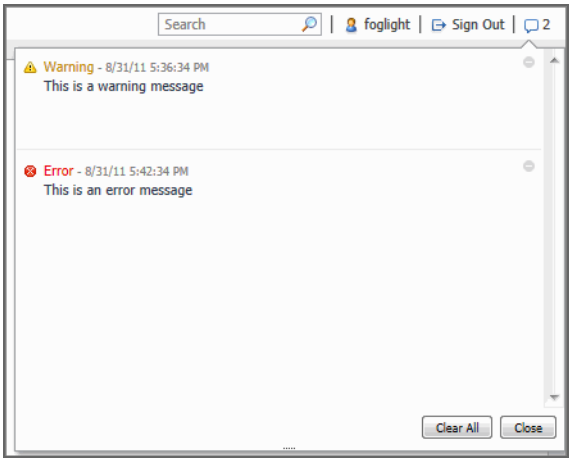
After a few moments, an Error message appears in the top-right corner.

Figure 5. Error Message



j Click to expand the list of messages.

Figure 6. List of messages



k **Optional**. Clear all or individual messages, or close the list.

Summary

In this tutorial you have learned to create Message Associations, Script Functions, Label, and Grid Layout views, and now to send messages to active Foglight® users.

Additional activities

After completing this tutorial, consider performing the following activities:

- Adding more messages
- Enhancing your Script Function to select a specific Message Association based on some selected criteria
- Call the *sendMessage()* function from your code after a process finishes executing

About Us

We are more than just a name

We are on a quest to make your information technology work harder for you. That is why we build community-driven software solutions that help you spend less time on IT administration and more time on business innovation. We help you modernize your data center, get you to the cloud quicker and provide the expertise, security and accessibility you need to grow your data-driven business. Combined with Quest's invitation to the global community to be a part of its innovation, and our firm commitment to ensuring customer satisfaction, we continue to deliver solutions that have a real impact on our customers today and leave a legacy we are proud of. We are challenging the status quo by transforming into a new software company. And as your partner, we work tirelessly to make sure your information technology is designed for you and by you. This is our mission, and we are in this together. Welcome to a new Quest. You are invited to Join the Innovation™.

Our brand, our vision. Together.

Our logo reflects our story: innovation, community and support. An important part of this story begins with the letter Q. It is a perfect circle, representing our commitment to technological precision and strength. The space in the Q itself symbolizes our need to add the missing piece—you—to the community, to the new Quest.

Contacting Quest

For sales or other inquiries, visit https://www.quest.com/company/contact-us.aspx.

Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at https://support.quest.com.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request.
- View Knowledge Base articles.
- Sign up for product notifications.
- Download software and technical documentation.
- View how-to-videos.
- Engage in community discussions.
- Chat with support engineers online.
- View services to assist you with your product.

Table 1. List of third-party contributions

Component	License or acknowledgment