



vFoglight™ 5.2.4

Web Component Guide



© 2008 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.

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

Quest Software World Headquarters

LEGAL Dept

5 Polaris Way

Aliso Viejo, CA 92656

www.quest.com

email: legal@quest.com

Refer to our Web site for regional and international office information.

Trademarks

Quest, Quest Software, the Quest Software logo, Aelita, Akonix, Akonix L7 Enterprise, Akonix L7 Enforcer, AppAssure, Benchmark Factory, Big Brother, DataFactory, DeployDirector, ERDisk, Foglight, Funnel Web, I/Watch, Imceda, InLook, IntelliProfile, InTrust, Invertus, IT Dad, I/Watch, JClass, Jint, JProbe, LeccoTech, LiteSpeed, LiveReorg, MessageStats, NBSpool, NetBase, Npulse, NetPro, PassGo, PerformaSure, Quest Central, SharePlex, Sitraka, SmartAlarm, Spotlight, SQL LiteSpeed, SQL Navigator, SQL Watch, SQLab, Stat, StealthCollect, Tag and Follow, Toad, T.O.A.D., Toad World, vANALYZER, vAUTOMATOR, vCONTROL, vCONVERTER, vEssentials, vFOGLIGHT, vOPTIMIZER, vRanger Pro, vReplicator, Vintela, Virtual DBA, VizionCore, Xaffire, and XRT are trademarks and registered trademarks of Quest Software, Inc in the United States of America and other countries. Other trademarks and registered trademarks used in this guide are property of their respective owners.

Disclaimer

The information in this document is provided in connection with Quest 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 products. EXCEPT AS SET FORTH IN QUEST'S TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, QUEST 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 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 HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Quest 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 does not make any commitment to update the information contained in this document.

License Credits and Third Party Information

To view license credit information, click the License Credits link on the Welcome to vFoglight online help page.

Web Component Guide
March 2009
Version 5.2.4

Table of Contents

| | |
|--|-----------|
| Introduction to this Guide | 7 |
| About vFoglight | 8 |
| About this Guide..... | 8 |
| vFoglight Documentation Suite | 9 |
| Core Documentation Set | 10 |
| Cartridge Documentation Sets | 10 |
| Feedback on the Documentation..... | 11 |
| Text Conventions | 11 |
| About Vizioncore Inc. | 12 |
| Contacting Dell | 14 |
| | |
| Introducing the Web Component Framework | 27 |
| Services Management with vFoglight..... | 28 |
| The vFoglight Browser Interface's Views | 28 |
| Configuring the Default Views | 28 |
| Overview of the Web Component Framework | 29 |
| The User Interface..... | 30 |
| Anatomy of a Typical Dashboard | 31 |
| View Components | 32 |
| | |
| Overview of Web Components | 43 |
| View Components, Containers, and Renderers | 44 |
| Common | 44 |
| Containers | 45 |
| Tables and Trees..... | 47 |
| Charts and Gauges | 47 |
| Time Range | 48 |

| | |
|--|-----------|
| Topology | 49 |
| Reporting | 49 |
| Inputs | 49 |
| Others | 51 |
| Renderers | 52 |
| The Web Component Framework | 55 |
| Core Concepts | 56 |
| Modules | 56 |
| Observations | 58 |
| Context | 62 |
| Parameters | 63 |
| On Null Values | 63 |
| Renderers | 63 |
| Default Values | 64 |
| Data Sources, Data Types, and Data Objects | 64 |
| Paths | 65 |
| Using the Web Component Framework | 66 |
| The Web Component Framework Editor | 66 |
| An Example Page | 68 |
| Web Component Framework in vFoglight | 69 |
| Managing Dashboards | 69 |
| Definitions Panes | 70 |
| Data and Data Sources Pages | 71 |
| Definitions Pane | 72 |
| Web Component Framework Workflow | 74 |
| Additional Documentation | 74 |
| Customizing the UI Quickly | 74 |
| Finding Pages: Bookmarks | 74 |
| Queries | 75 |
| Overview of Query Definitions | 76 |
| Creating a Query in vFoglight | 76 |
| Query Definition Settings | 77 |
| Conditional Types | 87 |

| | |
|---|------------|
| Sequence of Evaluation | 91 |
| Parameters in Queries | 94 |
| Creating a Query | 94 |
| Configuring Views and Context..... | 101 |
| Configuring Views | 102 |
| Creating a New Container View | 102 |
| Definitions Page for a View | 105 |
| Definitions Pane Settings Tabs | 107 |
| General Tab | 108 |
| Roles | 112 |
| Context tab | 112 |
| Configuration Tab | 113 |
| Flow tab | 114 |
| Layout Tab | 121 |
| Views Tab | 122 |
| Context and the Context Tab | 122 |
| Context Tab | 123 |
| Context Types | 124 |
| Additional Context | 125 |
| Dynamic Context | 126 |
| Flow Context | 127 |
| Runtime Values | 129 |
| Configurable Properties and Runtime Values | 130 |
| Simple Types | 130 |
| Runtime Value Types | 131 |
| Details of each Runtime Value | 133 |
| Additional Components | 147 |
| Renderers | 148 |
| Setting Renderers, Icons and Units in Views | 148 |
| Tasks | 150 |
| Icons | 151 |
| Files | 152 |
| Types | 153 |

| | |
|--|------------|
| Units..... | 154 |
| Determining the Appropriate Renderer for a Runtime Value | 154 |
| Theme and Module Resources..... | 155 |
| WCFTHEME | 156 |
| WCFMODULE | 156 |
| WCFMODULETHEME | 156 |
| Printing..... | 156 |
| Web Browser Printing | 157 |
| PDF Generation | 157 |
| Reports | 158 |
| Remote Access to Views | 160 |
| Portlet..... | 160 |
| Google Gadget..... | 161 |
| SharePoint Web Part | 162 |
| Index..... | 163 |

Introduction to this Guide

This chapter provides information about what is contained in the *vFoglight Web Component Guide*. It also provides information about the vFoglight documentation suite and Vizioncore.

This chapter contains the following sections:

| | |
|---|----|
| About vFoglight | 8 |
| About this Guide | 8 |
| vFoglight Documentation Suite | 9 |
| Text Conventions | 11 |
| About Vizioncore Inc. | 12 |

About vFoglight

vFoglight helps IT organizations understand the virtual infrastructure by managing the relationships and interaction between all the components in the environment, including data centers, data stores, clusters, resource pools, hosts and virtual machines. With vFoglight, administrators can quickly determine the root-cause of an incident or problem, track virtual machine (VM) movements and understand their impact, and identify contention for resources between virtual machines.

About this Guide

This Web Component Guide provides information about the vFoglight command-line interface. You can use vFoglight commands to interface with different components of your monitoring environment instead of the browser interface.

This guide is intended for vFoglight System Administrators who want to use the vFoglight commands.

The Web Component Guide is organized as follows:

Chapter 1, About the Command-Line Interface—Explains the command-line syntax, lists vFoglight commands and introduces the command-line interface using a getting started approach. Read this chapter to get an overview of vFoglight commands and how to get started.

Chapter 2, Managing the vFoglight Management Server—Describes the commands that allow you to perform server-related operations through the command-line interface and provides detailed instructions on how to get started with those commands. It provides reference information on server-related commands along with usage examples. Use the server-related commands to perform a variety of tasks such as starting or stopping the vFoglight Management Server, upgrading the database, or managing encryption keys.

Chapter 3, Managing the vFoglight Agent Manager—Describes the commands that allow you to access the vFoglight Agent Manager through the command-line interface along with instructions on how to configure your environment to obtain access to the commands that allow you to start or stop the vFoglight Agent Manager, display version information, or manage JVM options. Use this chapter to find reference information on the commands for managing the vFoglight Agent Manager along with usage examples.

Chapter 4, Managing Agents, Cartridges and Metrics—Provides information about the **fglcmd** interface that contains commands for managing common vFoglight entities

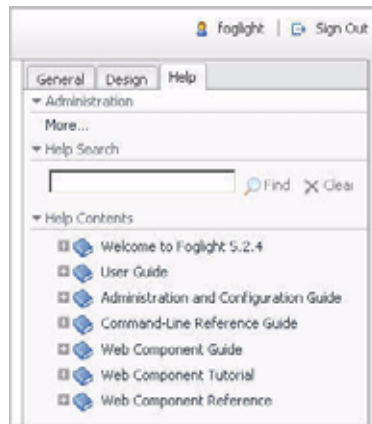
such as agents, cartridges and metrics. It also explains the **fglcmd** syntax and contains additional getting started instructions that show you how to configure your environment. Use this chapter to find reference information about the **fglcmd** commands and usage examples.

Appendix A, vFoglight Client Commands—Describes the commands that allow you to access the vFoglight Client through the command-line interface along with instructions on how to configure your environment to obtain access to the commands that allow you to start or stop the vFoglight Client display version information, or manage JVM options. Use this chapter to find reference information on the commands for managing the vFoglight Client along with usage examples.

vFoglight Documentation Suite

The vFoglight documentation suite is made up of the core documentation set, plus the documentation set for each vFoglight cartridge that you deploy. Documentation is provided in a combination of online help, PDF and HTML.

- **Online Help:** You can open the online help by selecting the Help tab from vFoglight's action panel.



- **PDF:** The *Getting Started Guide*, *What's New Guide*, *System Requirements and Platform Support Guide*, *Installation and Setup Guide* set, *Administration and Configuration Guide*, *vFoglight User Guide*, *Command-Line Reference Guide*, *Web Component Guide*, and *Web Component Tutorial*, are provided as PDF files.

The PDF guides are included in the zip file downloaded from Vizioncore. Adobe® Reader® is required.

- **HTML:** Release Notes are provided in HTML.

Core Documentation Set

The core documentation set consists of the following files:

- *Release Notes* (HTML)
- *Getting Started Guide* (PDF)
- *What's New Guide* (PDF)
- *System Requirements and Platform Support Guide* (PDF)
- *Installation and Setup Guide* set (all in PDF format):
 - Installation and Setup Guide—*Installing on Windows with an Embedded MySQL Database*
 - Installation and Setup Guide—*Installing on Windows with an External MySQL Database*
 - Installation and Setup Guide—*Installing on Windows with an External Oracle Database*
- *Administration and Configuration Guide* (PDF and online help)
- *vFoglight User Guide* (PDF and online help)
- *Advanced Configuration Guide* set
 - *Command-Line Reference Guide* (PDF and online help)
 - *Web Component Guide* (PDF and online help)
 - *Web Component Tutorial* (PDF and online help)
 - *Web Component Reference* (online help)

Cartridge Documentation Sets

When you deploy a cartridge, the documentation set for the cartridge is installed. The online help for the cartridge is integrated automatically with the core vFoglight help. When you open the help, the name of the cartridge is displayed in a top level entry within the table of contents.

Some cartridges include additional PDF guides, which may be one or more of the following: a *Getting Started Guide*, an *Installation Guide*, a *User Guide*, and a *Reference Guide*.

Feedback on the Documentation

We are interested in receiving feedback from you about our documentation. For example, did you notice any errors in the documentation? Were any features undocumented? Do you have any suggestions on how we can improve the documentation? All comments are welcome. Please submit your feedback to the following email address:

info@vizioncore.com

Please do not submit Technical Support related issues to this email address.

Text Conventions

The following table summarizes how text styles are used in this guide:

| Convention | Description |
|---|--|
| Code | Monospace text represents code, code objects, and command-line input. This includes: <ul style="list-style-type: none">• Java language source code and examples of file contents• Classes, objects, methods, properties, constants, and events• HTML documents, tags, and attributes |
| <i>Variables</i> | Monospace-plus-italic text represents variable code or command-line objects that are replaced by an actual value or parameter. |
| Interface | Bold text is used for interface options that you select (such as menu items) as well as keyboard commands. |
| <i>Files, components, and documents</i> | Italic text is used to highlight the following items: <ul style="list-style-type: none">• Pathnames, file names, and programs• The names of other documents referenced in this guide |

About Vizioncore Inc.

Vizioncore was formed in July 2002 as a consulting and software-development company with the mission to create easy-to-use software solutions that performed reliable and repeatable automation of datacenter functions specifically for the Citrix platform. A main corporate goal was to enable business partners to offer solutions that targeted real-world IT issues and provided the best possible installation and automation for their clients' systems.

Vizioncore's solutions have proved successful in organizations from small to mid-sized businesses to large enterprises, in a wide variety of vertical industries, including Financial Services, Government, Healthcare, Manufacturing, and High Tech. Vizioncore, Inc. can be found in offices around the globe and at www.vizioncore.com.

Contacting Dell

Note: If you do not have an active Internet connection, you can find contact information on your purchase invoice, packing slip, bill, or Dell product catalog.

Dell provides several online and telephone-based support and service options. Availability varies by country and product, and some services may not be available in your area. To contact Dell for sales, technical support, or customer service issues:

- 1 Visit <http://support.dell.com>.
- 2 Verify your country or region in the Choose A Country/Region drop-down menu at the bottom of the page.
- 3 Click Contact Us on the left side of the page. Note: Toll-free numbers are for use within the country for which they are listed.
- 4 Select the appropriate service or support link based on your need.
- 5 Choose the method of contacting Dell that is convenient for you.

| Country (City) | Service Type | Area Codes, Local Numbers, and Toll-Free Numbers Web and E-Mail Addresses |
|---|--|---|
| International Access Code Country Code City Code | | |
| Anguilla | Web Address E-Mail Address Technical Support., Customer Service, Sales | www.Dell.com/ai la-techsupport@dell.com toll-free: 800-335-0031 |
| Antigua and Barbuda | Web Address E-Mail Address Technical Support., Customer Service, Sales | www.Dell.com.ag la-techsupport@dell.com 1-800-805-5924 |
| Aomen | Technical Support Dell™ Dimension™, Dell Inspiron™, Dell Optiplex™, Dell Latitude™, and Dell Precision™ Servers and Storage | 0800-105 0800-105 |
| Argentina (Buenos Aires) International Access Code: 00 Country Code: 54 City Code: 11 | Web Address E-Mail Address for Desktop/ Portable Computers E-Mail Address for Servers and EMC® Storage Products Customer Service Technical Support Technical Support Services Sales | www.dell.com.ar la-techsupport@dell.com la_enterprise@dell.com toll-free: 0-800-444-0730 toll-free: 0-800-444-0733 toll-free: 0-800-444-0724 0-800-444-3355 |
| Aruba | Web Address E-Mail Address Technical Support., Customer Service, Sales | www.Dell.com/aw la-techsupport@dell.com toll-free: 800-1578 |
| Australia (Sydney) International Access Code: 0011 Country Code: 61 City Code: 2 | Web Address Contact Dell Web Address Technical Support., Customer Service, Sales | support.ap.dell.com support.ap.dell.com/contactus 13DELL-133355 |

| | | |
|--------------------------------|--|--|
| Austria (Vienna) | Web Address | Support.euro.dell.com |
| International Access Code: 900 | E-Mail Address | Tech_support_central_europe@dell.com |
| Country Code: 43 | Home/Small Business Sales | 0820 240 530 00 |
| City Code: 1 | Home/Small Business Fax | 0820 240 530 49 |
| | Home/Small Business Customer Service | 0820 240 530 14 |
| | Home/Small Business Support | 0820 240 530 17 |
| | Preferred Accounts/Corporate Customer | 0820 240 530 16 |
| | Service Preferred Accounts/Corporate Customer | 0820 240 530 17 |
| | Switchboard | 0820 240 530 00 |
| Bahamas | Web Address | www.dell.com/bs |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support., Customer Service, Sales | toll-free: 1-866-874-3038 |
| Barbados | Web Address | www.dell.com/bb |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support., Customer Service, Sales | 1-800-534-3142 |
| Belgium (Brussels) | Web Address | Support.euro.dell.com |
| | General Support | 02 481 92 88 |
| | General Support Fax | 02 481 92 95 |
| | Customer Service | 02 713 15 65 |
| | Corporate Sales | 02 481 91 00 |
| | Fax | 02 481 91 99 |
| | Switchboard | 02 481 91 00 |
| Bolivia | Web Address | www.dell.com/bo |
| | E-Mail Address | la_techsupport@dell.com |
| | Technical Support., Customer Service, Sales | toll-free: 800-10-0238 |
| Brazil | Web Address | www.dell.com/br |
| International Access Code: 00 | E-Mail Address | BR_TechSupport@dell.com |
| Country Code: 55 | Customer Service and Tech Support | 0800 970 3355 |
| City Code: 51 | Technical Support Fax | 51 2104 5470 |
| | Customer Service Fax | 51 2104 5480 |
| | Sales | 0800 722 3498 |
| British Virgin Islands | Technical Support, Customer Service, Sales | toll-free: 1-866-278-6820 |
| Brunei | Technical Support (Penang, Malaysia) | 604 633 4966 |
| Country Code: 673 | Customer Service (Penang, Malaysia) | 604 633 4888 |
| | Transaction Sales (Penang, Malaysia) | 604 633 4955 |
| Canada (North York, Ontario) | Online Order Status Web Address | www.dell.ca/ostatus |
| International Access Code: 011 | AutoTech (automated Hardware and Warranty Support) | support.ca.dell.com |
| | Customer Service | toll-free:1-800-247-9362 |
| | Home/Home Office | toll-free:1-800-847-4096 |
| | Small Business | toll-free:1-800-906-3355 |
| | Medium/Large Business, Government, Education | toll-free:1-800-387-5757 |
| | Hardware Warranty Phone Support | |
| | Computers for Home/Home Office | toll-free:1-800-847-4096 |
| | Computers for Small/Medium/Large Business | toll-free:1-800-387-5757 |
| | Government | |
| | Printers, Projectors, Televisions, Handheld, Digital | 1-877-335-5767 |
| | Jukebox, and Wireless Sales | toll-free:1-800-999-3355 |
| | Home and Home Office Sales | toll-free:1-800-387-5752 |
| | Small Business | toll-free:1-800-387-5755 |
| | Medium/Large Business, Government | 1 866 440 3355 |
| | Spare Parts and Extended Service | |
| Cayman Islands | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 1-877-262-5415 |

| | | |
|-------------------------|--|--|
| Chile (Santiago) | Web Address | www.dell.com/cl |
| Country Code: 56 | E-Mail Address | la-techsupport@dell.com |
| City Code: 2 | Sales and Customer Support | toll-free: 1230-020-4823 |
| China (Xiamen) | Technical Support Web Address | support.dell.com.cn |
| Country Code: 86 | Technical Support E-Mail Address | support.dell.com.cn/email |
| City Code: 592 | Customer Service E-Mail Address | customer_cn@dell.com |
| | Technical Support Fax | 592 818 14350 |
| | Technical Support – Dimension and Inspiron | toll-free: 800 858 2969 |
| | Technical Support – OptiPlex, Latitude and Dell Precision | toll-free: 800 858 0950 |
| | Technical Support – Servers and Storage | toll-free: 800 858 0960 |
| | Technical Support – Projectors, PDAs, Switches, Routers, etc | toll-free: 800 858 2920 |
| | Technical Support – Printers | toll-free: 800 858 2311 |
| | Customer Service | toll-free: 800 858 2060 |
| | Customer Service Fax | 592 818 1308 |
| | Home and Small Business | toll-free: 800 858 2222 |
| | Preferred Accounts Division | toll-free: 800 858 2557 |
| | Large Corporate Accounts GCP | toll-free: 800 858 2055 |
| | Large Corporate Accounts Key Accounts | toll-free: 800 858 2628 |
| | Large Corporate Accounts North | toll-free: 800 858 2999 |
| | Large Corporate Accounts North Government and Education | toll-free: 800 858 2955 |
| | Large Corporate Accounts East | toll-free: 800 858 2020 |
| | Large Corporate Accounts East Government and Education | toll-free: 800 858 2669 |
| | Large Corporate Accounts Queue Team | toll-free: 800 858 2572 |
| | Large Corporate Accounts South | toll-free: 800 858 2355 |
| | Large Corporate Accounts West | toll-free: 800 858 2811 |
| | Large Corporate Accounts Spare Parts | toll-free: 800 858 2621 |
| Columbia | Web Address | www.dell.com/co |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 01-800-915-4755 |
| Costa Rica | Web Address | www.dell.com/cr |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 0800-012-0231 |
| Czech Republic (Prague) | Web Address | support.euro.dell.com |
| International Access | E-Mail Address | czech_dell@dell.com |
| Code: 00 | Technical Support | 22537 2727 |
| Country Code: 420 | Customer Service | 22537 2707 |
| | Fax | 22537 2714 |
| | Technical Fax | 22537 2728 |
| | Switchboard | 22537 2711 |
| Denmark (Copenhagen) | Web Address | Support.euro.dell.com |
| International Access | Technical Support | 7023 0182 |
| Code: 00 | Customer Service – Relational | 7023 0184 |
| Country Code: 45 | Home/Small Business Customer Service | 3287 5505 |
| | Switchboard – Relational | 3287 1200 |
| | Switchboard Fax – Relational | 3287 1201 |
| | Switchboard – Home/Small Business | 3287 5000 |
| | Switchboard Fax – Home/Small Business | 3287 5001 |
| Dominica | Web Address | www.dell.com/dm |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-866-278-6821 |

| | | |
|--|--|--|
| Dominican Republic | Web Address | www.dell.com/do |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 1-800-156-1588 |
| Ecuador | Web Address | www.dell.com/ec |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales (Calling from Quito) | toll-free: 999-119-877-655-3355 |
| | Technical Support, Customer Service, Sales (Calling from Guayaquil) | toll-free: 1800-999-119-877-655-3355 |
| El Salvador | Web Address | www.dell.com/sv |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 800-6132 |
| Finland (Helsinki) | Web Address | support@euro.dell.com |
| International Access Code: 990 Country Code: 358 City Code: 9 | E-Mail Address | fi_support@dell.com |
| | Technical Support | 0207 533 555 |
| | Customer Service | 0207 533 538 |
| | Switchboard | 0207 533 533 |
| | Sales under 500 employees | 0207 533 540 |
| | Fax | 0207 533 530 |
| | Sales over 500 employees | 0207 533 533 |
| | Fax | 0207 533 530 |
| France (Paris) (Montpellier) International Access Code: 00 Country Code: 33 City Codes: (1) (4) | Web Address | Support.euro.dell.com |
| | Home and Small Business | |
| | Technical Support | 0825 387 270 |
| | Customer Service | 0825 832 833 |
| | Switchboard | 0825 004 700 |
| | Switchboard (calls from outside of France) | 04 99 75 40 00 |
| | Sales | 0825 004 700 |
| | Fax | 0825 004 701 |
| | Fax (calls from outside of France) | 04 99 75 40 01 |
| | Corporate | |
| | Technical Support | 0825 004 719 |
| | Customer Service | 0825 338 339 |
| | Switchboard | 55 94 71 00 |
| | Sales | 01 55 94 71 00 |
| Germany (Frankfurt) | Web Address | support.euro.dell.com |
| | E-mail Address | tech_support_central_europe@dell.com |
| | Technical Support | 069 9792-7200 |
| | Home/Small Business Customer Service | 0180-5-224400 |
| | Global Segment Customer Service | 069 9792-7320 |
| | Preferred Accounts Customer Service | 069 9792-7320 |
| | Large Accounts Customer Service | 069 9792-7320 |
| | Public Accounts Customer Service | 069 9792-7320 |
| Switchboard | 069 9792-7000 | |
| Greece | Web Address | Support.euro.dell.com |
| | Technical Support | 00800-44 14 95 18 |
| | Gold Service Technical Support | 00800-44 14 00 83 |
| | Switchboard | 2108129810 |
| | Gold Service Switchboard | 2108129811 |
| | Sales | 2108129800 |
| | Fax | 2108129812 |
| Grenada | Web Address | www.dell.com/gd |
| | E-Mail Address | la-techsuppo@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-866-540-3355 |

| | | |
|--|--|--|
| Guatemala | Web Address | www.dell.com/gt |
| | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 1-800-999-0136 |
| Guyana | E-Mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-877-270-4609 |
| Hong Kong | Web Address | support.ap.dell.com |
| International Access Code: 001 Country Code: 852 | Technical Support E-mail Address | support.dell.com.cn/email |
| | Technical Support - Dimension and Inspiron | 00852-2969 3188 |
| | Technical Support - OptiPlex, Latitude, and Dell Precision | 00852-2969 3191 |
| | Technical Support - Servers and Storage | 00852-2969 3196 |
| | Technical Support - Projectors, PDAs, Switches, Routers, etc . | 00852-3416 0906 |
| | Customer Service | 00852-3416 0910 |
| | Large Corporate Accounts | 00852-3416 0907 |
| | Global Customer Programs | 00852-3416 0908 |
| | Medium Business Division | 00852-3416 0912 |
| | Home and Small Business Division | 00852-2969 3105 |
| India | Dell Support Website | support.ap.dell.com |
| Portable and Desktop Support | | |
| | Desktop Support E-mail Address | india_support_desktop@dell.com |
| | Portable Support E-mail Address | india_support_notebook@dell.com |
| | Phone Numbers | 080-25068032 or 080-25068034 or your city STD code + 60003355 or toll-free: 1-800-425-8045 |
| Server Support | | |
| | E-mail Address | india_support_Server@dell.com |
| | Phone Numbers | 080-25068032 or 080-25068034 or your city STD code + 60003355 or toll-free: 1-800-425-8045 |
| Gold Support Only | | |
| | E-mail Address | eec_ap@dell.com |
| | Phone Numbers | 080-25068033 or your city STD code + 60003355 or toll-free: 1-800-425-9045 |
| Customer Service | | |
| | Home and Small Business | India_care_HSB@dell.com toll-free : 1800-4254051 |
| | Large Corporate Accounts | India_care_REL@dell.com toll free : 1800-4252067 |
| Sales | | |
| | Large Corporate Accounts | 1600 33 8044 |
| | Home and Small Business | 1600 33 8046 |

| | | |
|-------------------------------|--|--|
| Ireland (Cherrywood) | Web Address | Support.euro.dell.com |
| International Access Code: 00 | Technical Support | |
| Country Code: 353 | E-mail Address | dell_direct_support@dell.com |
| City Code: 1 | Business computers | 1850 543 543 |
| | Home computers | 1850 543 543 |
| | At Home Support | 1850 200 889 |
| | Sales | |
| | Home | 1850 333 200 |
| | Small Business | 1850 664 656 |
| | Medium Business | 1850 200 646 |
| | Large Business | 1850 200 646 |
| | E-mail Address | Dell_IRL_Outlet@dell.com |
| | Customer Service | |
| | Home and Small Business | 204 4014 |
| | Business (greater than 200 employees) | 1850 200 982 |
| | General | |
| | Fax/Sales fax | 204 0103 |
| | Switchboard | 204 4444 |
| | U.K. Customer Service (dealing with U.K. only) | 0870 906 0010 |
| | Corporate Customer Service (dial within U.K. only) | 0870 907 4499 |
| | U.K. Sales (dial within U.K. only) | 0870 907 4000 |
| Italy (Milan) | Web Address | Support.euro.dell.com |
| International Access Code: 00 | Home and Small Business | |
| Country Code: 39 | Technical Support | 02 577 826 90 |
| City Code: 02 | Customer Service | 02 696 821 14 |
| | Fax | 02 696 821 13 |
| | Switchboard | 02 696 821 12 |
| | Corporate | |
| | Technical Support | 02 577 826 90 |
| | Customer Service | 02 577 825 55 |
| | Fax | 02 575 035 30 |
| | Switchboard | 02 577 821 |
| Jamaica | E-mail Address | ja-techsupport@dell.com |
| | Technical Support, Customer Service, Sales (dial from within Jamaica only) | 1-800-440-920 |

| | | |
|--|--|------------------------------|
| Japan (Kawasaki) | Web Address | support.jp.dell.com |
| International Access Code: 001 Country Code: 81 City Code: 44 | Technical Support - Dimension and Inspiron | toll-free: 0120-198-26 |
| | Technical Support outside of Japan - Dimension and Inspiron | 81-44-520-1435 |
| | Technical Support - Dell Precision, OptiPlex, and Latitude | toll-free: 0120-198-433 |
| | Technical Support outside of Japan - Dell Precision, OptiPlex, and Latitude | 81-44-556-3894 |
| | Technical Support - Dell PowerApp™, Dell PowerEdge™, Dell PowerConnect™, and Dell PowerVault™, | toll-free: 0120-198-498 |
| | Technical Support outside of Japan - PowerApp, PowerEdge, PowerConnect, and PowerVault | 81-44-556-4162 |
| | Technical Support - Projectors, PDAs, Printers, Routers | toll-free: 0120-981-690 |
| | Technical Support outside of Japan - Projectors, PDAs, Printers, Routers | 81-44-556-3468 |
| | Faxbox Service | 044-556-3490 |
| | 24-Hour Automated Order Status Service | 044-556-3801 |
| | Customer Service | 044-556-4240 |
| | Business Sales Division - up to 400 employees | 044-556-1465 |
| | Preferred Accounts Division Sales - over 400 employees | 044-556-3433 |
| | Public Sales - government agencies, educational institutions, and medical institutions | 044-556-5963 |
| Global Segment Japan | 044-556-3469 | |
| Individual User | 044-556-1657 | |
| Individual User Online Sales | 044-556-2203 | |
| Individual User Real Site Sales | 044-556-4649 | |
| Switchboard | 044-556-4300 | |
| Korea (Seoul) | Web Address | Support.ap.dell.com |
| International Access Code: 001 Country Code: 82 City Code: 2 | Technical Support, Customer Service | toll-free: 080-200-3800 |
| | Technical Support - Dimension, PDA, Electronics, and Accessories | toll-free: 080-200-3801 |
| | Sales | toll-free: 080-200-3600 |
| | Fax | 2194-6202 |
| | Switchboard | 2194-6000 |
| Latin America | Customer Technical Support (Austin, Texas, U.S.A.) | 512 728-4093 |
| | Customer Service (Austin, Texas, U.S.A.) | 512 728-3619 |
| | Fax (Technical Support and Customer Service) (Austin, Texas, U.S.A.) | 512 728-3883 |
| | Sales (Austin, Texas, U.S.A.) | 512 728-4397 |
| | SalesFax (Austin, Texas, U.S.A.) | 512 728-4600 or 512 728-3772 |
| Luxemborg | Web Address | Support.euro.dell.com |
| International Access Code: 00 Country Code: 352 | Support | 3420808075 |
| | Home/Small Business Sales | +32 (0)2 713 15 96 |
| | Corporate Sales | 26 25 77 81 |
| | Customer Service | +32 (0)2 481 91 19 |
| | Fax | 26 25 77 82 |
| Macao | Technical Support | toll-free: 0800 105 |
| Country Code: 83 | Customer Service (Xiamen, China) | 34 160 910 |
| | Transaction Sales (Xiamen, China) | 29 693 115 |

| | | |
|-------------------------------|--|--|
| Malaysia (Penang) | Web Address | Support.ap.dell.com |
| International Access Code: 00 | Technical Support - Dell Precision, OptiPlex, and Latitude | toll-free: 1800 880 193 |
| Country Code: 60 | Technical Support - Dimension, Inspiron, and Electronics and Accessories | toll-free: 1800 881 306 |
| City Code: 4 | Technical Support - PowerApp, PowerEdge, PowerConnect, and PowerVault | toll-free: 1800 881 386 |
| | Customer Service | toll-free: 1800 881 306 (option 6) |
| | Transaction Sales | toll-free: 1800 888 202 |
| | Corporate Sales | toll-free: 1800 888 213 |
| Mexico | Web Address | www.dell.com/mx |
| International Access Code: 00 | E-mail Address | la-techsupport@dell.com |
| Country Code: 52 | Customer Technical Support | 001-877-384-8979 or 001-877-269-3383 |
| | Sales | 50-81-8800 or 01-800-888-3355 |
| | Customer Service | 001-877-384-8979 or 001-877-269-3383 |
| | Main | 50-81-8800 or 01-800-888-3355 |
| Montserrat | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | Toll-free: 1-866-278-6822 |
| Netherlands | E-mail Address | la-techsupport@dell.com |
| Antilles | Web Address | support.euro.dell.com |
| Netherlands (Amsterdam) | Technical Support | 020 674 45 00 |
| International Access Code: 00 | Technical Support Fax | 020 674 47 66 |
| Country Code: 31 | Home/Small Business Customer Service | 020 674 42 00 |
| City Code: 20 | Relational Customer Service | 020 674 43 25 |
| | Home/Small Business Sales | 020 674 55 00 |
| | Relational Sales | 020 674 50 00 |
| | Home/Small Business Sales Fax | 020 674 47 75 |
| | Relational Sales Fax | 020 674 47 50 |
| | Switchboard | 020 674 50 00 |
| | Switchboard Fax | 020 674 47 50 |
| New Zealand | Web Address | Support.ap.dell.com |
| International Access Code: 00 | E-mail Address | Support.ap.dell.com/contactus |
| Country Code: 64 | Technical Support, Customer Service, Sales | 0800 441 567 |
| Nicaragua | Web Address | www.dell.com/ni |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 001-800-220-1377 |
| Norway (Lysaker) | Web Address | Support.euro.dell.com |
| International Access Code: 00 | Technical Support | 671 16882 |
| Country Code: 47 | Relational Customer Service | 671 17575 |
| | Home/Small Business Customer Service | 231 62298 |
| | Switchboard | 671 16800 |
| | Fax Switchboard | 671 16865 |
| Panama | Web Address | www.dell.com/pa |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 011-800-507-1264 |
| Peru | Web Address | www.dell.com/pe |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 0800-50-669 |

| | | |
|-----------------------------------|--|--|
| Poland (Warsaw) | Web Address | support.euro.dell.com |
| International Access Code: 011 | E-mail Address | pl_support_tech@dell.com |
| Country Code: 48 | Customer Service Phone | 57 95 700 |
| City Code: 22 | Customer Service | 57 95 999 |
| | Sales | 57 95 999 |
| | Customer Service Fax | 57 95 806 |
| | Reception Desk Fax | 57 95 998 |
| | Switchboard | 57 95 999 |
| Portugal | Web Address | Support.euro.dell.com |
| International Access Code: 00 | Technical Support | 707200149 |
| Country Code: 351 | Customer Service | 800 300 413 |
| | Sales | 800-300-410 or 800-300 -411 or 800-300-412 or 21-422-07-10 |
| | Fax | 21-424-01-12 |
| Puerto Rico | Web Address | www.dell.com/pr |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 1-877-537-3355 |
| St. Kitts and Nevis | Web Address | www.dell.com/kn |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-866-540-3355 |
| St. Lucia | Web Address | www.dell.com/lc |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-866-464-4352 |
| St. Vincent and the Grenadines | Web Address | www.dell.com/vc |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-866-464-4353 |
| Singapore | NOTE: The phone numbers in this section should be called from within Singapore or Malaysia only. | |
| International Access Code: 005 | Web Address | support.ap.dell.com |
| Country Code: 65 | Technical Support - Dimension, Inspiron, and Electronics and Accessories | toll-free: 1 800 394 7430 |
| | Technical Support - OptiPlex, Latitude, and Dell Precision | toll-free: 1 800 394 7488 |
| | Technical Support - PowerApp, PowerEdge, PowerConnect, and PowerVault | toll-free: 1 800 394 7478 |
| | Customer Service | toll-free: 1 800 394 7430 (option 6) |
| | Transaction Sales | toll-free: 1 800 394 7412 |
| | Corporate Sales | toll-free: 1 800 394 7419 |
| Slovakia (Prague) | Web Address | support.euro.dell.com |
| International Access Code: 00 | E-mail Address | czech_dell@dell.com |
| Country Code: 421 | Technical Support | 02 5441 5727 |
| | Customer Service | 420 22537 2707 |
| | Fax | 02 5441 8328 |
| | Tech Fax | 02 5441 8328 |
| | Switchboard (Sales) | 02 5441 8328 02 5441 7585 |
| South Africa (Johannesburg) | Web Address | support.euro.dell.com |
| International Access Code: 09/091 | E-mail Address | dell_za_suppor@dell.com |
| Country Code: 27 | Gold Queue | 011 709 7713 |
| City Code: 11 | Technical Support | 011 709 7710 |
| | Customer Service | 011 709 7707 |
| | Sales | 011 709 7700 |

| | | |
|--------------------------------|--|--|
| Spain (Madrid) | Web Address | Support.euro.com |
| International Access Code: 00 | Home and Small Business | |
| Country Code: 34 | Technical Support | 902 100 130 |
| City Code: 91 | Customer Service | 902 118 540 |
| | Sales | 902 118 541 |
| | Switchboard | 902 118 541 |
| | Fax | 902 118 539 |
| | Corporate | |
| | Technical Support | 902 100 130 |
| | Customer Service | 902 115 236 |
| | Switchboard | 91 722 92 00 |
| | Fax | 91 722 95 83 |
| Sweden (Upplands Vasby) | Web Address | support.euro.dell.com |
| International Access Code: 00 | Technical Support | 08 590 05 199 |
| Country Code: 46 | Relational Customer Service | 08 590 05 642 |
| City Code: 8 | Home/Small Business Customer Service | 08 587 70 527 |
| | Employee Purchase Program (EPP) Support | 020 140 14 44 |
| | Technical Support Fax | 08 590 05 594 |
| Switzerland (Geneva) | Web Address | Support.euro.dell.com |
| International Access Code: 00 | E-mail Address | Tech_support_central_Europe@dell.com |
| Country Code: 41 | Technical Support – Home and Small Business | 0844 811 411 |
| City Code: 22 | Technical Support – Corporate | 0844 822 844 |
| | Customer Service – Home and Small Business | 0848 802 202 |
| | Customer Service – Corporate | 0848 821 721 |
| | Fax | 022 799 01 90 |
| | Switchboard | 022 799 01 01 |
| Taiwan | Web Address | support.ap.dell.com |
| International Access Code: 002 | E-mail Address | support.dell.com.cn/email |
| Country Code: 886 | Technical Support - OptiPlex, Latitude, Inspiron, Dimension, and Electronics and Accessories | toll-free: 0080 186 1011 |
| | Technical Support - Servers and Storage | toll-free: 0080 160 1256 |
| | Customer Service | toll-free: 0080 160 1250 (option 5) |
| | Transaction Sales | toll-free: 0080 165 1228 |
| | Corporate Sales | toll-free: 0080 165 1227 |
| Thailand | Web Address | Support.ap.dell.com |
| International Access Code: 001 | Technical Support (OptiPlex, Latitude, and Dell Precision) | toll-free: 1800 0060 07 |
| Country Code: 66 | Technical Support (PowerApp, PowerEdge, PowerConnect, and PowerVault) | toll-free: 1800 0600 09 |
| | Customer Service | toll-free: 1800 006 007 (option 7) |
| | Corporate Sales | toll-free: 1800 006 009 |
| | Transaction Sales | toll-free: 1800 006 006 |
| Trinidad/Tobago | Web Address | www.dell.com/ff |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-888-799-5908 |
| Turks and Caicos Islands | Web Address | www.dell.com/tc |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-877-441-4735 |

| | | |
|--------------------------------|---|--|
| U.K.(Bracknell) | Web Address | upport.euro.dell.com |
| International Access Code: 00 | E-mail Address | dell_direct_support@dell.com |
| Country Code: 44 | Customer Service Website | support.euro.dell.com/uk/en/ECare/form/home.asp |
| City Code: 1344 | Sales | |
| | Home and Small Business Sales | 0870 907 4000 |
| | Corporate/Public Sector Sales | 01344 860 456 |
| | Customer Service | |
| | Home and Small Business | 0870 906 0010 |
| | Corporate | 01344 373 185 |
| | Preferred Accounts (500-5000 employees) | 0870 906 0010 |
| | Global Accounts | 01344 373 186 |
| | Central Government | 01344 373 196 |
| | Local Government & Education | 01344 373 199 |
| | Health | 01344 373 194 |
| | Technical Support | |
| | Corporate/Preferred Accounts/PCA (1000+ employees) | 0870 908 0500 |
| | Other Dell Products | 0870 353 0800 |
| | General | |
| | Home and Small Business Fax | 0870 907 4006 |
| Uruguay | Web Address | www.dell.com/uy |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 000-413-598-2521 |
| U.S.A. (Austin, Texas) | Automated Order-Status Service | toll-free: 1-800-433-9014 |
| International Access Code: 011 | AutoTech (portable and desktop computers) | toll-free: 1-800-247-9362 |
| Country Code: 1 | Hardware and Warranty Support (Dell TV, Printers, and Projectors) for Relationship customers | toll-free: 1-877-459-7298 |
| | Consumer (Home and Home Office) Support for Dell products | toll-free: 1-800-624-9896 |
| | Customer Service | toll-free: 1-800-624-9897 |
| | Employee Purchase Program (EPP) Customers | toll-free: 1-800-695-8133 |
| | Financial Services Web Address | www.dellfinancialservices.com |
| | Financial Services (lease/loans) | toll-free: 1-877-577-3355 |
| | Financial Services (Dell Preferred Accounts [DPA]) | toll-free: 1-800-283-2210 |
| | Business | |
| | Customer Service | toll-free: 1-800-624-9897 |
| | Employee Purchase Program (EPP) | toll-free: 1-800-695-8133 |
| | Customer s Support for printers, projectors, PDAs, and MP3 players | toll-free: 1-877-459-7298 |
| | Public (government, education, and healthcare) | |
| | Customer Service and Support | toll-free: 1-800-456-3355 |
| | Employee Purchase Program (EPP) Customers | toll-free: 1-800-695-8133 |
| | Dell Sales | toll-free: 1-800-289-3355 or toll-free: 1-800-879-3355 |
| | Dell Outlet Store (Dell refurbished computers) | toll-free: 1-888-798-7561 |
| | Software and Peripherals Sales | toll-free: 1-800-671-3355 |
| | Spare Parts Sales | toll-free: 1-800-357-3355 |
| | Extended Service and Warranty Sales | toll-free: 1-800-247-4618 |
| | Fax | toll-free: 1-800-727-8320 |
| | Dell Services for the Deaf, Hard-of-Hearing, or Speech-Impaired | toll-free: 1-877-DELLTTY (1-877-335-5889) |

| | | |
|---------------------|--|--|
| U.S. Virgin Islands | Web Address | www.dell.com/vi |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | toll-free: 1-877-702-4360 |
| Venezuela | Web Address | www.dell.com/ve |
| | E-mail Address | la-techsupport@dell.com |
| | Technical Support, Customer Service, Sales | 0800-100-4752 |

Introducing the Web Component Framework

The *Web Component Framework* (WCF) is the software that enables you to build a browser interface and perform specific tasks such as monitoring data. By configuring these views, you can display data in a variety of tabular and graphical formats. The retrieved data can be filtered, sorted, and truncated. The full list of properties for each component is given in the view pages that are accessible from the **Help** menu on the browser interface. This document provides an introduction to these components and describes the underlying mechanisms that allow them to display data retrieved from vFoglight or other sources with the same data structure.

For a quick introduction to how dashboards are built and populated with sample views, try the Vizioncore View Component Tutorial, which is also accessible from the **Help** menu on the vFoglight browser interface.

This chapter provides information about dashboards and the components used to build them.

The WCF is not just a tool to build a plain Web page. With it you can build pages that update themselves, you can add drill down pages that depend on the context of the choice made on the parent page, and add a large number of useful components to the pages that you build. This manual describes the WCF and shows you how to understand and use it.

This chapter contains the following sections:

| | |
|---|----|
| Services Management with vFoglight | 28 |
| Overview of the Web Component Framework | 29 |
| The User Interface | 30 |

Services Management with vFoglight

Enterprise services management applications typically gather gigabytes of monitoring data and then attempt to organize the data in a meaningful way. That's 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 drilldown pages to increasingly specific information about a chosen component, such as a single host or a particular database instance.

The top-level screens, those you see when the browser interface is first launched, have been organized around the concept of services and have been designed 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's views you will imagine ways that they could be improved to better fit the way that you would like to organize and visualize your data.

The vFoglight Browser Interface's Views

Vizioncore's designers anticipated a user's need to customize the browser interface, so they included the means to allow you to access the UI's component framework and with it to create custom views. You can populate these views with other display components, such as charts and tables, and connect them to data sources. It is the same data that the vFoglight agents have been configured to collect, but now it is organized in a way that best fits a given business model and the information needs particular to that model.

The end result is a monitoring system that organizes data in a way that mirrors the business model. Real-time monitoring data is presented in a way that is easily viewed, and it fosters better control of the application's availability. This also helps with service level management. Because custom views show services in a cleaner way, a monitor 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

Configuring the Default Views

vFoglight employs a configurable Web-based interface. By doing your own custom configurations, you apply your detailed knowledge of your system to augment or replace the out-of-the-box views vFoglight shows by default.

You can modify the existing components in the browser interface using the following operations:

- Create a custom dashboard easily by dragging existing views or data from the action panel to it. This is the simplest way of creating a new view.

You can add any view that is designated as a portlet, thus building up a custom page. The data tab on the action panel presents choices from which you can drag metric charts and position them on the page.

- In the action panel, you can adjust the width of the views place on a page by choosing the number of columns.
- Add dashboards to My Definitions. This requires more expertise, but it allows access to the framework, so you can define completely new views.
- Create a Report.
- Add Bookmarks.

Overview of the Web Component Framework

The *Web Component Framework* provides the underlying structure from which you can build the user interface for your application. A configuration framework makes it easy to deploy dashboards and their views into various application environments. The components it requires for operation are accessed through an interface so that the application that uses the *Web Component Framework* can provide these services from their own infrastructure without having to use arbitrary mechanisms that are not core to the *Web Component Framework*.

The *Web Component Framework* consists of a structure for hosting related views called view components, and container services that host data sources. It is a superset 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.

The *Web Component Framework* is written in Java and is capable of running in a web container such as 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.

It supports multiple data sources. With it you can configure multiple data queries and display this data using views. Queries are the primary mechanism used to extract data from the *Web Component Framework* data sources. A view can use more than one query (to extract data for display), and a query can use more than one data object.

Apart from creating applications using the *Web Component Framework*, there are several important considerations:

- The Dashboard interface (default views)
- Data interface (data representation, relationships in the system, ability to query data)
- Persistence interface (data storage)
- Permissions interface (ability to set rules and privileges).

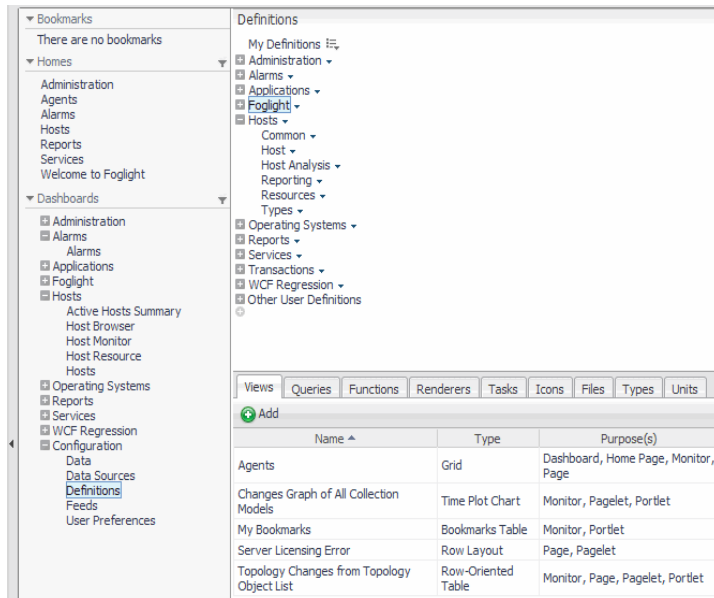
See the vFoglight core documentation set for a discussion of these topics.

The User Interface

The *vFoglight User Guide* describes the overall appearance of the vFoglight user interface. This document describes the part of the interface used to define, view and edit elements of the *Web Component Framework*. Other top-level trim elements contain buttons supplied by the application. This manual concerns itself with the Definitions area that is accessed in the right navigation panel. The Definitions area allows you to examine and work with all the existing entities in the *Web Component Framework* that is used to build all the views in the vFoglight user interface.

The figure shows that you access the Definitions area by choosing **Definitions** under **Dashboards > Configuration** in the left navigation panel.

Note The *Design* tab is available on any non-portal page to those whose role includes that of dashboard designer. It shows a hierarchical list of all the views on the page. Selecting a particular view shows information about it in the lower pane, allowing you easy access to child views.



The vFoglight user interface consists of top-level view components called dashboards that contain other user interface elements. These components can respond to user actions, refreshing a view or drilling down to other, more detailed views with dynamic content.

Anatomy of a Typical Dashboard

- A dashboard consists of a container, which in turn contains views.
- Views are assembled from Vizioncore View Components.
- View components are often required to display specific data, which may be set at design time or may be based on user interaction, which demands, in addition to query-based data retrieval, a mechanism for passing information in the form of parameters.
- In the *Web Component Framework*, the query mechanism allows you to retrieve data from a datasource, for example, vFoglight data from a vFoglight data source.
- Queries can be parameterized, which gives them an extra degree of flexibility in the dynamic data that they retrieve.
- A flow mechanism permits pages to be updated or linked to other pages.

- 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 *Web Component Framework* components is the Definitions choice. You may find it useful to have vFoglight open on the Definitions page so that you can refer to it as you continue reading.

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, and reports
- Data visualization components, such as charts, tables, gauges, labels, and trees
- Specialized components, such as RSS feeds

The configuration settings include flow control, contextual inputs, data binding, and query specification. Views can be configured using:

- A query—to provide data binding.
- A context—what is shown in the view depends on the context passed to the page or component.
- A flow—an action to be performed based on user input, such as a selection of a particular item on the parent page that launches a drill down page based on the context.

The *vFoglight User Guide* covers the following topics in greater detail.

Pages can be decorated with headers, which may contain:

- Breadcrumbs—the present page name preceded by other page names
- Time Region—which may contain
 - Timestamp: if no time range is applied to the page
 - Time Range: if available and applies to all views
 - A Zonar: if available, permits a choice of the time interval
 - Nothing: if multiple time ranges are represented
- Optional Page Scope Actions, such as

- Time Range: change the time range for components on the page

The right action panel contains:

- General tab:
 - Page actions, such as set properties, create a bookmark, set as home page, and print the view
 - Other actions, where you can create a new, empty dashboard, which is simultaneously a portal, into which you can drag existing views as a way of quickly customizing a new page, and where you can create a report
 - Themes, where you can choose one of the existing themes for the user interface
- Design tab:
 - The views contained in the parent view
 - The definition, layout, and context of the selected view
- Help tab:
 - Help for all components on the current page
 - The online help documentation set

Configuring a Component—an Example

Before explaining in detail view components, their configurable properties, runtime values, and context settings, an example may help to show the various options that are available to the designer who wants to configure a component. Because of its simplicity, the chosen component is a Label.

The *Web Component Framework* editor in vFoglight is the tool used to configure this component, which is accessed starting from the left navigation panel by choosing **Dashboards > Configuration > Definitions**, and then My Definitions in the Module List pane.

These choices cause the Module Contents pane, which is below the Module List Pane, to show all the views that have been defined for the current user. If this user has not yet created any views, the pane will be empty.

For example, here are the steps required to configure the Label component in the editor pane.

To configure a Label:

- 1 Ensure that the **Views** tab in the Module Contents pane is selected.

- 2 Click the **Add** button.
The *New View* dialog opens.
- 3 Click **Common > Label** in the **Blank view** drop-down list, and then click **OK**.
The Module Definitions pane changes to an editor pane.
- 4 Only the component's name and its **Label** property are required to have a functioning component. The next few sections give a brief description of each tab in the Definitions Editor.

The General Tab

The settings on the General tab are described in “[General Tab](#)” on page 108, but for now just note that fields with required settings are marked with an exclamation icon (❗), while others can retain their default values. The label must be given a name and a size. It should be given a description in the Comments field that gives an indication of why it is being used. Because this simple component will in all probability be added to a page, its purpose should be set to Pagelet. Its roles, which are matched to user roles to control who is allowed to see the component, need not be considered in this introductory example. When this page is filled in, the component has a name, a preferred size, a description, some allowed roles, and some purpose or purposes.

The Context Tab

In the *Web Component Framework*, context is analogous to the environment information available to applications running under the control of an operating system. Context is the collection of available information that can be copied and passed on to the component. This information may be needed by the component to properly fulfill its purpose.

The Configuration Tab

The Configuration tab is where the label's properties are set. A primary goal of this example is to give an overview of the possibilities available to users to bind data to a component, so the next section discusses these possibilities as they apply to each of the label's properties.

Binding Data to a Component

To blend well with other components on a page, a view component needs to have configurable properties that control its appearance. To be useful, a view component must have mechanisms for data binding. The *Web Component Framework* supports dynamic configuration for both appearance and data display.


Data Types

Each property has a data type. By demanding that all data have an assigned type the *Web Component Framework* can perform checks at design time to reduce the risk of incompatible assignments.

Setting the Properties for a Label

Property: Label

The editor shows that the data type for this component is *Any*, which means that you have complete freedom in assigning any type of data to this component. Of course, the system must be able to convert whatever it is given to a visible string or the label won't be of much use.

You set the data type by clicking the **Edit** icon () and choosing one of the available types, which are listed in “[Runtime Value Types](#)” on page 131.

Some of the available runtime value types are:

- **String**

For the label component, the first choice is String. This is a simple type that permits you to define a static string value for the label.

- **Rich Text**

The next choice is Rich Text that permits you to add HTML tags, so that you can do some simple formatting, like italicizing a word.

- **Context Selection**

The next choice is context selection, and here there is a richer choice of options, as shown in the dialog:

Context Selection Runtime Value Show Advanced

Input Key ...

Treat as Type ...

Path ...

Return First Object in List

Renderer ...

Return Type ▾

Unit Property Name ▾

Time Range to Use ...

| Name | Value |
|---------|-------|
| On Null | |

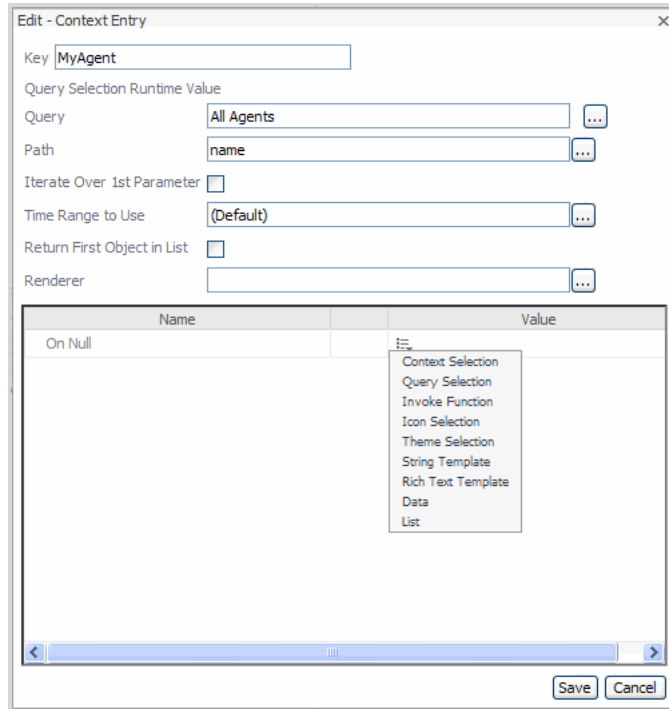
Save Cancel

The required item is a key, which indicates that a context entry must be declared before it can be used. You can extract information from the context, which was perhaps set in a parent view or in the Additional section of the Label's **Context** tab, and display it as the label's string. Thus, the label's text can be dynamic and be made to depend on a user's choice at run time.

- **Query Selection**

The chief way of extracting results from the data model at run time is by using a query, which returns a list of data objects. Often, the list contains a single data object, and you can select the specific piece of information you want in that object and use it as the label's display string. There are many options for using a query selection, including one for the case where the query returns an empty list. The accompanying figure shows that the same choices for selecting the data apply to this case too.

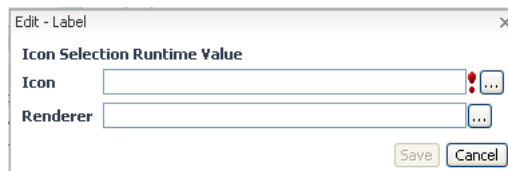
The Query Selection dialog:



- **Icon Selection**

An Icon is a collection of related images of different sizes. You can use one of these images in the label property. The dialog allows you to choose an image that you have previously added and a renderer for that image.

For more information, see “[Icon Selection](#)” on page 138.



- **Theme Selection**

Themes allow the selection of different colors for a background, so that if the theme is switched the selected color can be changed as well. The edit dialogs for

themes allow you to specify the theme's properties. For more information, see [“Theme Selection”](#) on page 137.

Dialog box titled "Edit - Label" showing the "Theme Selection Runtime Value" section. It contains three input fields: "Component", "Style", and "Value". At the bottom right are "Save" and "Cancel" buttons.

- **String Template**

A string template allows you to type in the value for the label and if necessary to supply a renderer. Its purpose is to allow you to combine different data elements in a chosen order.

For more information, see [“String Template”](#) on page 139.

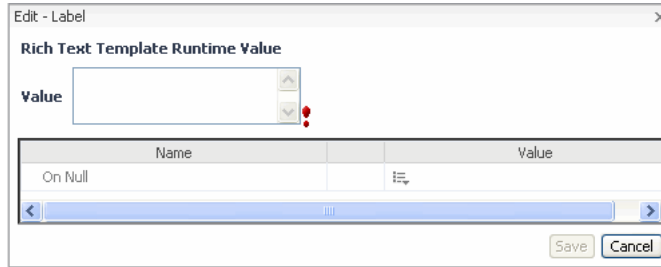
Dialog box titled "Edit - Label" showing the "String Template Runtime Value" section. It includes a "Value" field with a dropdown arrow, a "Renderer" field with an ellipsis button, and a table with columns "Name" and "Value". The table has one row with "On Null" in the "Name" column. At the bottom right are "Save" and "Cancel" buttons.

| Name | Value |
|---------|-------|
| On Null | |

- **Rich Text Template**

A rich text type permits HTML formatting. Its value is text with simple HTML formatting tags. It accepts positional parameters of the type {0}, {1}, and so on. The Rich Text type permits HTML formatting, but no positional parameters.

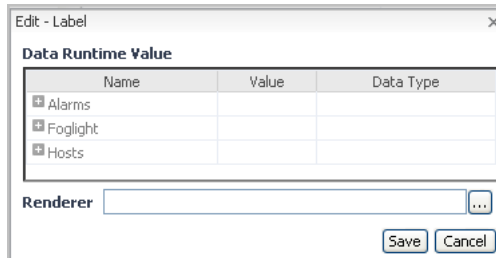
For more information, see [“Rich Text and Rich Text Template”](#) on page 140.



- **Data**

The data choice is for end users who expect to have the same type of information available whenever vFoglight is run. The dialog presents a chooser showing the data types and data values available in the current vFoglight instance.

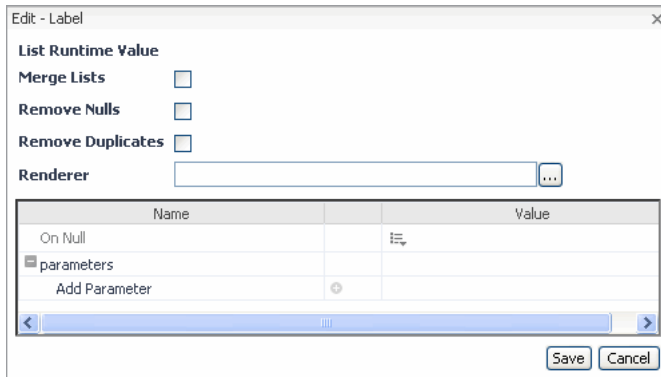
For more information, see “[Data](#)” on page 141.



- **List**

In the list data choice, a parameter is used to select the list. This allows you to choose the parameterized list of items.

For more information, see “[List](#)” on page 142.

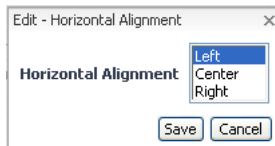


Property: Item

Item specifies an object that is made available as a context entry to the flow actions. A flow is an action that is triggered by clicking the label. A special dynamic generated context whose key is *Item* is made available. Dynamic context is information added to the context that sets its value based on user interaction. In this case perhaps the user has clicked an item in a list displayed by the label component. That particular choice is recorded in the context and is made available to a view that appears as a result of a flow action defined on the label.



Property: Horizontal Alignment



There are three choices, Left, Center, and Right.

Property: No Wrap

This is a boolean property that controls the way that a long line of text appears in the label.

Property: Title

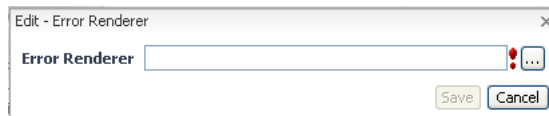
Title has the same choices for its content as Item. The title appears as a header in the bounding box around the label, but only if you choose to show it when adding the label to its parent view.

Property: Background

This group of properties controls various aspects of the label's background. See the reference pages for additional details.

Property: Error Renderer

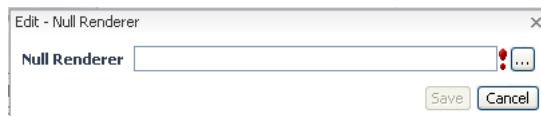
Choose one of the available error renderers.



An error renderer is invoked if the retrieval of the specified data generates an error at runtime.

Property: Null Renderer

Choose one of the available null renderers.



A null renderer is invoked if the retrieval of the specified data returns null.

Flow

The Flow tab allows you to specify the actions that can be performed when a user clicks the label or just hovers over it.

For more information, see “[Configuring Flows for Views](#)” on page 115.

There are additional properties under **Page Options** and **Portlet Options** that you can look up in the reference page for the label.

From this example you can see that even a simple component such as a label has many configurable properties and actions. This overview of the possible settings for a Label should give you a notion of how you would use the definitions editor to set properties on a *Web Component Framework* entity.

Once you have configured some components you can add them to a container such as a Grid, and in this way you can build a complex view containing a variety of entities for displaying the data that vFoglight collects.

Overview of Web Components

The full list of View components and their containers have properties given in the [Vizioncore Web Component Reference](#). Refer to it for the complete list of properties for each component. A synopsis of what each container and view component does or is used for is given here as a quick reference. It may help you to get an overview of the available components and aid you to make the choice of an appropriate component.

This chapter contains the following sections:

[View Components, Containers, and Renderers](#)44

View Components, Containers, and Renderers

The list of user interface components is shown in the following tables.

Common

| | |
|---------------------------|--|
| Drop-Down List | Permits the selection of an object from a list of Runtime Value objects. |
| Image | Displays a specified image and allows for interaction. |
| Key-Value Listing | Displays a table of keys (labels), values, and optionally associated states that are usually all properties of one object, such as a Host or an AppServer. |
| Label | Permits the display a single piece of information. |
| Row-Oriented Table | See Tables and Trees . |
| Time Plot Chart | See Charts and Gauges . |
| Time Range Zonar | See Time Range . |

Containers

These layouts are available:

| | |
|----------------------|---|
| Column Layout | Displays a set of views organized in columns. Within each column, the views are positioned vertically in the same way as the views in a Grid layout, except that each column is laid out independently. The natural height of the layout is based on the tallest column, and the natural width is the sum of the column widths. (Each column's natural width is based on the widest view in that column.) |
| Fixed Layout | Uses absolute (x, y) locations and fixed sizes to precisely locate all of its contained views. All views are exactly placed and sized. Useful for monitoring views with fixed real estate when you want to make sure that all the important information is always on the screen. |
| Form | Provides the frame for views that act as forms. It allows you to define one or more submit actions, which are rendered as buttons using the standard look and feel. |
| Grid | Presents the views it contains in a zero-based grid of cells. |
| Iterator | Presents a single view repeatedly for each object in a list. The Iterator supplies the context required by the individual view page or pages. |
| Report | Contains a series of views that are rendered, in the order in which they appear in the layout (except for header and footer Page Decorations, which must precede the body views), to a ServerReport document for printing or saving as a PDF. |
| Row Layout | Is similar to a Column Layout, except that it organizes a set of views in rows. The width of a cell in one row is independent of those in the row above or the row below. |
| Splitter | Takes two views and displays them within the space of the layout, separated by a movable bar. |
| Stack Layout | Layers a set of views on top of one another. All views are adjusted to the same size, and are drawn from back to front. |

| | |
|----------------------|---|
| Switch | Takes a single object or list of objects, usually specified by a context selection, as an input. This layout displays one child view at a time based on an input. It matches the type of the input, and whether or not it is a list, against the list of child views and then displays the child whose type most closely matches the input. In other words, the input is matched against the child views in the order that they are defined; in this way a more general view (accepting of a broader type) can be listed after a number of more specifically defined views and act as the “default” case. |
| Tab Container | Shows the currently selected view in a set of views that have been added to a menu. The current view may be changed by selecting a different view from the menu bar. All views are maintained, and participate in context updates. |
| Topology | Allows you to define different views of up to four zoom levels for combinations of Data Type and Is List, where Is List controls whether the display is for a single object of that Data Type or a list of them. |
| Type | Is similar in appearance to the Switch layout, but the view that is displayed for a given input object is based on the set of all views that accept an object of that type as a sole input. The major configuration property for a type-based layout is the input property, an object whose type is used to determine the view to display. |
| Wizard Layout | Displays a group of views in a set sequence that should facilitate a workflow. Validation may be added as well. |

Tables and Trees

| | |
|----------------------------|---|
| Array Table | A standard table view that lists data in rows and columns. |
| Cell-Oriented Table | Places data in individual cells of a table. |
| Key-Value Listing | Displays a table of keys (labels), values, and optionally associated states, that are usually all properties of one object, such as a Host or an AppServer. |
| Row-Oriented Table | Consists of a list of rows in which there are groups of columns. Each group consists of a list of columns. |
| Tree Table | Consists of list of groups; each group consists of a list of columns. |

Charts and Gauges

| | |
|--------------------------|---|
| Bar Gauge | Displays the current value of each one of the configured metrics as a bar. |
| Bar or Pie Chart | Graphs data items as a bar chart (with each bar representing one item), a stacking bar, or a pie chart. |
| Chart Legend | Shows the visual attributes (symbol, line, or fill style) of each data series in a parent chart. |
| Circular Gauge | Generates a PNG image of a circular gauge, for example, a speedometer or temperature dial on a boiler. |
| Cluster Bar Chart | Graphs data values retrieved from parent data objects as clusters of bars. |
| Cylinder | Generates a PNG image of a cylinder. The gauge displays one-dimensional data as a filled percentage of the container. |

| | |
|-------------------------|--|
| Pulse Gauge | Generates an animated GIF of an arrow that travels in one direction across the width or height of the GIF. |
| Time Bar Chart | Graphs metric data as a bar chart against time. The <i>x</i> -axis represents time; the <i>y</i> -axis represents the metric. |
| Time Plot Chart | Graphs metric data as a plot line against time. The <i>x</i> -axis represents time; the <i>y</i> -axis represents the metric. |
| Time Range Zonar | See Time Range . |
| TimeState Chart | Graphs discrete state data (such as availability or trend data) as a series of symbols against time. Only for enum-based data. |

Time Range

| | |
|-----------------------------|---|
| Time Range Drop-Down | <p>Use this component when you need:</p> <ul style="list-style-type: none"> • A list of time ranges that allows you to choose intervals from ‘last hour’ to ‘all time’. • An icon that allows you to freeze the current time range so that updating a page or its children does not shift the time range. Data remains static until you revert to real time once more. • To select more than one time range on the same page and have each selection affect only certain views on that page. |
| Time Range Form | Allows you to specify a time range by selecting dates in a calendar. |
| Time Range Zonar | Allows you to select a time range by dragging a shaded rectangle along the time axis. |

Topology

| | |
|-------------------|---|
| Minuscule | For use in the Minuscule zoom level of the Topology layout. It either displays an icon (via a State Renderer) or displays nothing, but makes a desired color available to an enclosing layout (which may render it as a small box). |
| Title Only | Commonly used in Topology layouts, at the Minimized zoom level. It does not display anything by itself, but allows you to configure a <i>Title</i> for the component, which will be rendered by whatever layout the component is included in, as long as that layout is set up to render the title. |
| Topology | Renders the objects in the topology tree for the data model as a graph of nodes. Allows you to define different views for up to four zoom levels for combinations of Data Type and Is List, that is, whether for a single object of that Data Type or a list of them. |

Reporting

| | |
|------------------------|---|
| Page Decoration | Used to define headers and footers in a report layout. |
| Report | Contains a series of views that are rendered in sequence (in the order they appear in the layout) to a ServerReport document for printing or saving as a PDF. |

Inputs

| | |
|------------------------------|--|
| Button | A clickable button. |
| Check Box | A check box with an attached label and optional title. |
| Context Inputs Editor | This component is not intended for end users. It allows the setting and changing of a isolated set of context variables. |

| | |
|-----------------------------|--|
| Date/Time Input | Allows the user to assign a Java date object and output it as Date or Time. |
| Drop-Down List | Permits the selection of an object from a list of Runtime Value objects. |
| Filter | Allows the user to filter a list of objects, for example, a list of Events, on the basis of matching certain properties. |
| Number Input | Allows the input of Java-typed numbers, such as integer, double, or float. It also enables you to set the range of the input value, using the <i>minValue</i> and <i>maxValue</i> properties. |
| Radio Button List | Displays a list of runtime values, such as a list of hosts, with a selection button for each and permits you to pick one <i>Runtime Value</i> from the list. |
| Text Area | Allows the input of multiple lines of Java-typed strings that are composed solely of visible characters. |
| Text Field | Allows the input of multiple lines of Java-typed strings that are composed solely of visible characters. |
| Time Range Drop-Down | For cases in which: <ul style="list-style-type: none">• You want a customized list of time ranges that is different from the global list of time ranges.• You want to be able to select more than one time range on the same page and have each selection affect only certain views on that page. |
| Time Range Form | Allows you to specify a time range by selecting dates in a calendar. |

Others

| | |
|-------------------------|--|
| Filter | Allows you to filter a list of objects, for example, a list of Events, on the basis of matching certain properties. |
| iFrame | Allows the user to embed an HTML page into the <i>Web Component Framework</i> so that the page can pass parameters to a target URL. |
| Include | Includes the resource specified in the view configuration. The resource can be dynamic or static. |
| Links Box | Displays a list of clickable links which can be used to configure navigation for a page or drive actions on the page. |
| List Viewer | This component does not have a description in the help pages. It is for internal use only. |
| Page Title | A component whose Text property can serve as a title for a page containing other views. |
| Progress | Displays a progress bar to which actions can be wired. Shows the progress of your tasks either with detailed progress or a busy indicator. |
| Property Viewer | This component does not have a description in the help pages. It is for internal use only. |
| Separator | The Separator component is used to display a vertical or horizontal line. |
| Syndication Feed | Allows you to publish a list of Data Objects as a syndication feed. |
| Toolbar | Displays a row of buttons that fire actions when clicked. |
| Wizard Layout | Displays a group of views in a set sequence that should facilitate a workflow. Validation may be added as well. |

Renderers

| | |
|-----------------------|--|
| Date | Displays Date objects in a locale-specific way. |
| Error | The message to display when an error occurs. |
| Host Name | Takes a string that is presumably a host name and trims off the number of address sections that it is configured to remove. The default trim segment count should be 2. Other than that, it is just a normal String Renderer. |
| Icon | Looks up the Icon registered to the type of the object being rendered and extracts and renders an appropriately sized image from the Icon. Icons are registered to data types in the Type Mapping entity. Various sizes of images (from <i>extraSmall</i> to <i>huge</i> , plus a scalable image) are set in the configuration of the Icon entity. |
| List | A List renderer is used to display a list of values. For each of values, the List renderer attempts to choose an appropriate renderer for the data type. (Examples: <i>StringRenderer</i> for String, <i>NumberRenderer</i> for Number). |
| Null Image | Specifies the icon to use when a null condition in a data binding occurs. |
| Null String | Specifies the text to use when a null condition in a data binding occurs. |
| Number | Converts numeric data to text or HTML. How the data is formatted depends on the values set in the renderer's properties. |
| Number Bar | Used to render either a numeric value as a sideways bar or a list of numeric values as a sideways stacking/cluster bar. |
| Number Percent | Displays a numeric value and then a sideways number bar that shows a graphical representation of the percentage. |

| | |
|-----------------------------|---|
| Number Unit | Formats numeric data with an accompanying unit to text or HTML. How the data is formatted depends on the values set in the renderer's properties. All properties of the Number renderer apply to the Number Unit renderer. |
| Pulse Gauge Renderer | A simple renderer that generates an animated GIF of an arrow that travels in one direction across the width or height of the GIF. |
| Range | Associates number values with an Icon (with sizing), a Name, and a Color. |
| Sparkline | A handy way of showing the trend of a metric in a small space. When used in a table it provides easy shape comparison. |
| State | Associates state values with an Icon (with sizing), a Name, and a Color. |
| String | Allows you to configure the appearance of the output string. |
| Threshold | Shows the history of a metric in terms of the thresholds the historic average falls under. It uses the default state renderer for the threshold enumeration to choose a color for the given threshold, thus rendering the history as a colored bar. For each section of the bar, a title tag provides information in words about the value of that section. |
| Time Range | Displays a time range. |

The Web Component Framework

The vFoglight user interface is built using the Web Component Framework. This same framework is available to end users who wish to develop their own specialized views. Existing vFoglight views are configurable from the user interface without the need to resort to the full editing resources provided by the Web Component Framework, but if you decide to build your own specialized dashboards, the Web Component Framework is available for your use.

This chapter contains the following sections:

| | |
|---|----|
| Core Concepts | 56 |
| Managing Dashboards | 69 |

Core Concepts

Certain core concepts find application throughout the *Web Component Framework*. These core concepts are:

- [Modules](#)
- [Observations](#)
- [Context](#)
- [Parameters](#)
- [On Null Values](#)
- [Renderers](#)
- [Default Values](#)
- [Data Sources, Data Types, and Data Objects](#)
- [Paths](#)

Modules

The default *Web Component Framework* definitions (views, queries, renderers, tasks, icons, files, types, and units) in vFoglight are organized onto modules and sub-modules. A module contains a collection of related definitions. With the proper privileges you can add entities to any of the existing modules, and users with access to the Definitions choice (in the navigation panel under **Dashboards > Configuration**) can create entities in their own modules, which you can save by exporting them to a file. Also, you can import modules that have been created on another vFoglight server.

Importing and Exporting Modules

For information on importing and exporting modules, see the *Command-Line Reference Guide*.

Validating Modules

You can validate your module and check all the definitions within it for errors or warnings. This is useful for catching errors that might have crept in through hand coding or prior incompatibilities in the WCF tooling that allowed poorly-configured views to be saved. This functionality can be invoked from the module's menu.

Definitions and Entities

Definitions are the configurable units in *Web Component Framework*. They include Views, [Queries](#), [Tasks](#), [Icons](#), [Types](#) and [Units](#).

Entities are the types of definitions which can be referred to by an ID identifier in other definitions. Views, [Queries](#), [Tasks](#), [Icons](#) are entities.

Note Views and their properties are described in the Web Component Reference.

Public

Marking an Entity as public indicates that it can be used by any definition in any other module. For this reason the editors will not allow you to delete a public entity. However, you have the option of replacing a view containing a public entity.

Entities that are not public can only be referenced by definitions that have a shared module ancestry. Definitions have a shared module ancestor if they have a parent, grandparent, and so on, module that is the same or one is an ancestor of the other.

When selecting an entity, only public entities are shown in modules that don't have a common ancestry with the module containing the current definition.

Copying Entities

When copying entities, whether entities are public or not needs to be taken into consideration.

You can make a shallow copy of an entity that is not public. However, if that entity refers to a non-public entity, that is, refers to a query via a Query Selection Runtime Value, then the copy will not be allowed if that non-public entity is not accessible from the target module to which you want to save the copy.

The View's *deep copy* feature can be used to copy a view that references a non-public entity that is not accessible from the target module. However, a warning message will be issued that the inaccessible, non-public entity will be copied and the reference will be changed to refer to the copy. This may not be what you want. You may actually want the references to be maintained because you intend to deep copy a view to a temporary module, make the desired changes to the view and its related entities, and then deep copy the view back to its original module.

You can perform either a copy of the view or a deep copy of the view without having to use the Add View work flow to invoke the copy. Moreover, if you wish to do a simple copy, but are not permitted to because of private reference restrictions, then a deep copy

can optionally be invoked to minimize the duplication required to copy the specific view.

Observations

In general, the *Web Component Framework* and vFoglight are concerned with the collection of information over time. These collections are called observations. Observations can be of any kind of object. Observations that are handled specially by the *Web Component Framework* are Metrics and Enum Observations. The scope and quality of the information returned by an observation in the *Web Component Framework* is determined by the time range used to retrieve the information.

Time Range Related to Observations

A *TimeRange* for the a Metric Observation can be specified in various ways, but it is always ultimately composed of a range of date-time objects, and a granularity. The granularity can be RAW, which means that data observations should be shown in the metric history with the smallest available granularity, or a number of milliseconds, for example, 300,000 for 5 minutes. It can also be AUTO, for example, a numeric value of -2, meaning the code will pick the best granularity based on the time range.

The list of history Metric Values must be in ascending order of date-time.

When the granularity is RAW, indicated by the numeric value -1, history observations will be added with the smallest available granularity. Different observations within the history may cover differently-sized time intervals, and different metrics in the same time range using RAW granularity may have different numbers of history observations in the case of unrelated time intervals.

When the granularity is a number of milliseconds, in each history metric value, the difference between the end time and the start time (in milliseconds) must equal the granularity. Each successive history object's start time should be the previous history object's end time. That is, each history object must cover exactly one granularity interval.

Exactly how the start times and end times of the history objects are related to the start time and end time of the entire non-RAW time range is not specified as a *Web Component Framework* requirement. For one thing, the length of the entire time range may not be an even multiple of the granularity. Reasonable options include starting the first history object's start time at the exact start time of the time range, or starting it at the start time of the time range, less half the granularity. However, the start time of the

first history object should be no greater than the start time of the time range, and the end time of the last history object should be no less than the end time of the time range.

When there are multiple metrics calculated for the same non-RAW time range (whether they are different metrics on the same object, or the same metric on different objects), they must have exactly the same number of history objects as each other, and the start times and end times of the history objects at the same indices in their lists must be identical.

If there are no agent observations that correspond to a history object within a certain granularity interval in a non-RAW time range, a history object is still created, with the appropriate start time and end time, but none of the value properties are filled in.

Metric Observations

An observation of a numerical value is called a metric.

Data collected by agents is generally expected to be collected into a Metric data object. Certain components, such as the chart components, must be bound to Metric objects to function.

Metric definitions must specify the *unit* property in the Metric's containing Property class. The unit is used to display numeric data. In particular, the unit's *precision* property is used by some components (such as charts) and renderers to ensure the correct number of decimal places are used when rendering data. If the Metric's precision is desired to be different than its unit's precision, you can set the precision directly on the Metric's containing Property class using the *precision* property.

Metrics must conform to the following schema:

Metric

| Property | Type | Description |
|----------|--------------------------------|--|
| uniqueId | String | unique ID |
| name | String | name of metric, localizable |
| period | MetricValue | value for entire time range |
| current | MetricValue | value for last interval in time range |
| history | MetricValue; is-many = true | list of values for each interval in time range |

MetricValue

| Property | Type | Description |
|-------------------|-------------|--|
| uniqueId | String | unique id |
| startTime | Date | start time of interval, exclusive |
| endTime | Date | end time of interval, inclusive |
| sampledPeriod | Long | period covered by this interval in milliseconds. (Might be less than endTime - startTime if agent was not collecting data during the entire interval.) |
| count | Long | number of agent samples in this interval |
| min | Number | minimum value of all agent samples in this interval |
| max | Number | maximum value of all agent samples in this interval |
| average | Number | average value of all agent samples in this interval |
| sum | Number | sum of all agent samples in this interval |
| sumSquares | Number | sum of the squares of all agent samples in this interval |
| standardDeviation | Number | standard deviation of all agent samples in this interval |
| baselineMin | Number | baseline minimum value of all agent samples in this interval |
| baselineMax | Number | baseline maximum value of all agent samples in this interval |

| Property | Type | Description |
|------------|----------------------------------|--------------------------------------|
| thresholds | ThresholdValue is-many = true | list of thresholds for this interval |

ThresholdValue

| Property | Type | Description |
|------------|--------|--|
| uniqueId | String | unique id |
| upperBound | Bound | upper bound of threshold range |
| lowerBound | Bound | lower bound of threshold range |
| state | Enum | state associated with this threshold range |

Bound

| Property | Type | Description |
|-----------|---------|---|
| uniqueId | String | unique id |
| value | Number | value of <i>bound</i> . If set to <i>Double.POSITIVE_INFINITY</i> , the threshold is assumed to stretch to infinity |
| exclusive | Boolean | false if the bound includes value, true if it does not |

Enum Observations

Data collected by agents is sometimes collected into an *EnumObservation* data object. Certain components, such as the time state chart component, must be bound to *EnumObservation* objects to function. *EnumObservations* must conform to the following schema:

EnumObservation

| Property | Type | Description |
|----------|--------------------------------------|--|
| uniqueId | String | unique id |
| name | String | name of metric, localizable |
| period | EnumObservationValue | value for entire time range |
| current | EnumObservationValue | value for last interval in time range |
| history | EnumObservationValue; is-many = true | list of values for each interval in time range |

EnumObservationValue

| Property | Type | Description |
|-----------|--------|-----------------------------------|
| uniqueId | String | unique id |
| startTime | Date | start time of interval, exclusive |
| endTime | Date | end time of interval, inclusive |
| index | Number | Index of this observation |
| value | Number | Enum value of this observation |

Context

Context is the collection of data that defines the *subject* of the view. If a view is configured to display information about one application server, then the context specifies which application server.

If the context of a view has been set, then the view can use that value by specifying the named value of the context. Similarly, if a view has a given named value in its context,

and a link takes it to another view, then that new view can get access to that value by specifying its name as a context input.

For more information, see “[Context and the Context Tab](#)” on page 122.

Parameters

Most types of Runtime Values have parameters. These are placeholders nested within the Runtime Value that are also evaluated at run time. An example is a String Template Runtime Value set to *Host: {0}*. In this case *Host:* is a fixed string, while *{0}* is a reference to a parameter that evaluates to the name of the host server by configuring that association to a particular dynamic context at the time when the String Template Runtime Value is being defined. Each description of the Runtime Value types explains how to use parameters.

Caution Any quotation mark you use in the parameterized string must be escaped. For example, if the string that you want to use is *The host's name is: {0}*, you must escape the apostrophe as follows:
The host\'s name is: {0}.

For more information, see “[Configurable Properties and Runtime Values](#)” on page 130.

On Null Values

Most types of Runtime Value have an associated On Null Runtime Value. This provides an alternate value that can be used if the main Runtime Value evaluates to null (nothing) or an empty list of values.

Renderers

A renderer may be specified on each type of Runtime Value. A renderer determines how the evaluated value is displayed. For example, a limit to the length of a string, the number of decimal places, or the date and time format. If no renderer is specified, then default renderers are used. Therefore, in most cases, a renderer is not required.

Null and error renderers, noted earlier in the list of simple types, are special cases of renderers. They are only used to display no data or error information. Unlike normal renderers, they do not base their behavior on the data being rendered, but instead have fixed outputs. A drop-down menu lists all the available renderers. Generally, you can determine which renderer to select based on its name. If no renderer is specified,

renderers are looked up from the type, property and unit of the value. If no renderer is found, a default renderer is used. Therefore, in most cases, a renderer is not required.

For more information, see “[Renderers](#)” on page 148.

Default Values

If a property has a default value it is displayed as text beside the edit icon in the Value column of the tree table in the Configuration tab. A property that has been configured to use a specific value can be set back to its default property by clicking the **Edit** icon, then selecting **Clear to default** from the drop-down menu.

Data Sources, Data Types, and Data Objects

The data that is displayed by the application comes from a data source, as set up in vFoglight. The data from a data source is held in data objects as properties. For example, some of the Host data-object properties are the name of the host, its IP address, and the number of fatal events. The data type is a data-object template, and determines the structure of a data object. Examples of data types are Host, AppServer, WebLogic, WebSphere, Agent, and Event. For more information about each property's default, see `<install directory>\docs\module-catalogue.html`.

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.

You can investigate an instance of the graph of vFoglight data source objects by choosing Configure > Data in the browser interface.

Data Source Queries

- The syntax of a data source query resembles SQL, but queries return a list of objects.
- Data source queries are strongly typed.

- Data source queries are not free form. In vFoglight, they are configured on the browser interface's Definitions page by using an editor provided for that purpose.

Paths

When creating queries and defining Runtime Values, you set the values by specifying the data object and the properties in the Path field. Paths traverse the structure of the data object. They are similar to directory paths in Windows or UNIX, and are comprised of a series of one or more property names, separated by forward slashes.

Note You can replace a long path by setting it as a context value.

One minor complication is that property names are often displayed with localized names instead of their actual property names. For example, a drop-down tree may show the properties as Name or CPU Usage, but when selected they display in the Path field as name or *cpuUsage*.

The following are some example paths:

| Path | Meaning |
|----------------------|---|
| /hosts | An absolute path for all hosts, under the top of the tree of data. |
| cpuUsage | The cpuUsage metric object (under a host object) |
| cpuUsage/current/avg | The current average value of the cpuUsage metric object (under a host object) |

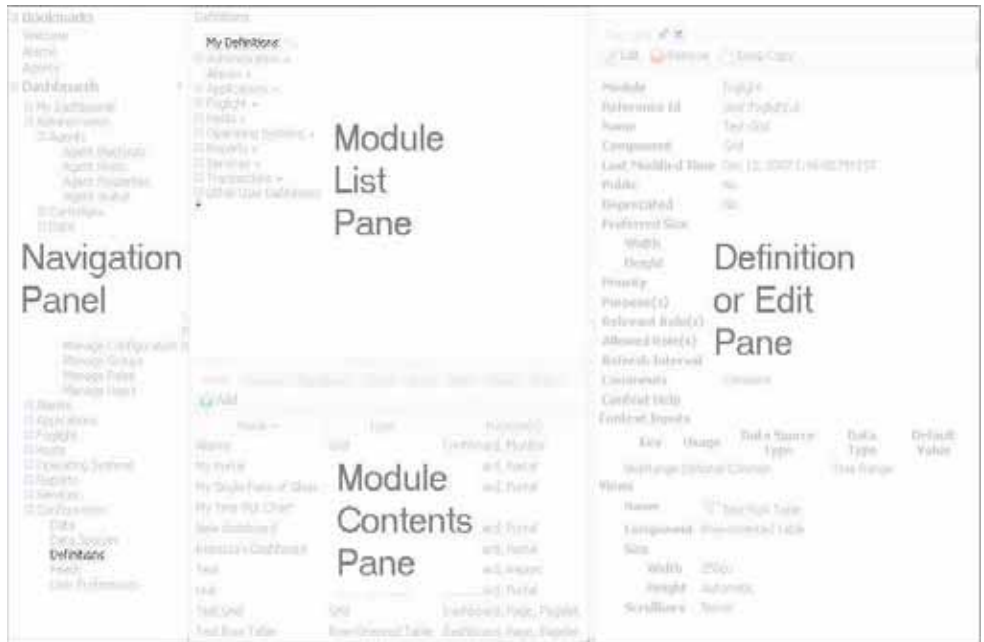
Using the Web Component Framework

The Web Component Framework Editor

You navigate to the editor by choosing **Configuration > Definitions** in the navigation panel. Both the navigation and action panels are described in the *vFoglight User Guide*.

The *Web Component Framework* editor consists of:

- A Module List pane that displays the existing modules. Most modules are defined by the application developer. You can add your views to the module called My Definitions, which is associated with your login name, although you can add your views anywhere you choose (if you have the proper permissions). Other users can add views as well. If such a user was creating views in his or her My Definitions workspace, you would locate these views under Other User Definitions, where the modules are once more arranged by login name.
- A Module Contents pane. This pane consists of eight tabs. Within each tab you can create the following entities for the selected module: views, queries, renderers, tasks, icons, files, types, and units.
- A Definitions or Edit pane. It functions as an editor pane, and as an item definition pane once the item has been configured. You use this pane to edit a new or existing definition. Once the definition is saved the pane switches to a view of the definition.



In vFoglight, you access these panes from **Dashboards > Configuration > Definitions** in the navigation panel.

An Example Page

Services (All Alarms) Tuesday, December 18, 2007 9:27 AM - Now 4.0 hours

Categories and Services Category Selector

| Name | Service Level Compliance | Alarms |
|---------------------------|--------------------------|--------|
| Applications | ● | |
| test2 | ● | |
| test | ● | |
| Hosts | ● | 2 |
| Windows | ● | 2 |
| tor014488.prod.quest.corp | ● | 2 |

Tasks

- Custom
- Last Hour
- Last 4 Hours
- Last 8 Hours
- Last 24 Hours
- Last 48 Hours
- Last 72 Hours
- Today
- Yesterday
- This Week
- Last Week
- This Month
- Last Month
- This Year
- Last Year
- All Time

2 : 2 Outstanding Alarm(s) for the Entire System (Not Including SLA Alarms)

View: 2 Outstanding Alarm(s) | 2 Alarm Source(s) | 1 Related Host(s) | 2 Related Agent(s)

Select All Unselect All Acknowledge Clear

| Sev | Time | Host | Source |
|-----|------------------|---------------------------|---|
| ● | 12/10/07 5:04 PM | tor014488.prod.quest.corp | FileSys_Table C: FileSystem C:: free |
| ● | 12/10/07 4:06 PM | tor014488.prod.quest.corp | AppMonitor_AppMonito... Agent "AppMonitor |

QUEST SOFTWARE Copyright Quest Software, Inc. Contact Us Allow

A page may be:

- A context-free dashboard
- A dependent page, which optionally requires a context
- A 2-pane browser: navigator and page

All of the preceding are assembled from view components.

The page may contain:

- A variable number of view components
- Page decorations, such as a time range control
- Tabs or a splitter, depending on the container being used
- A customizer, which is one way of linking to other pages

Views can be configured using:

- A query—to provide data binding
- A context—what is shown depends on the context passed to the page or component
- A flow—an action to be performed based on user input, such as a drill down page

Other data binding choices exist. For more information, see “[Configurable Properties and Runtime Values](#)” on page 130.

View components and their containers have properties that are described in the *Vizioncore Web Component Reference*. Refer to it for details about these properties.

Web Component Framework in vFoglight

The views in vFoglight are constructed from the UI components in the *Web Component Framework*. The data in the views come from the vFoglight data source. If you want to know what data is available in a running vFoglight instance, you can browse it in the Definitions > Data tab.

A user with the proper permissions can build new views to better address an organization’s specific needs. If you want to customize vFoglight by designing and building your own add-on views, a good place to become familiar with the process is the Vizioncore View Component Tutorial. It leads you through the basics of constructing new dashboards and dependent views and introduces you to some frequently-used components.

You can modify:

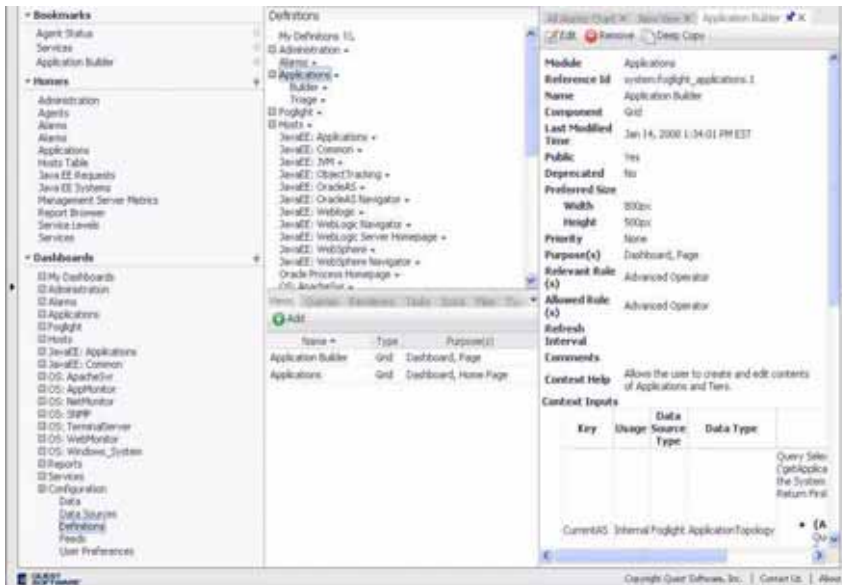
- a page by adding, removing or moving views
- a view, for example, by adding columns, removing metrics, and changing required input
- a query by adjusting the data to retrieve or the criteria with which to select it.

Managing Dashboards

The following section describes the Dashboard, Definitions, and Data Sources pages.

Definitions Panes

From the Module List pane, accessed in the navigation panel under *Configuration > Definitions*, which lists all the available, currently-defined dashboards, you can examine the dashboards that have been defined for all the System and User modules. After familiarizing yourself with the available dashboards, you can decide whether a custom one needs to be built, and you can perhaps copy and modify one of these instead of creating one from scratch.



System and User Dashboards

In vFoglight, there are two types of dashboards: system and user. System dashboards are preconfigured and delivered with vFoglight. User dashboards are created or modified by users. Depending on your permissions, System dashboards may or may not be displayed in the Dashboards page that you see.

vFoglight comes with preconfigured views, queries, renderers, tasks, icons, files, types, and units. Users can view and copy these definitions, and with the proper permissions they can change them as well. Alternatively, you can create and modify your own User modules. Similar to System modules, you can view and copy modules created by another user but you cannot modify those definitions.

As was mentioned previously, you can copy and make modifications to an existing view, or you can create an entirely new view. The choice depends upon how similar to an existing one the new page must be. These remarks apply to queries as well. It is better to copy and modify a complex query rather than recreating it if only a few changes are required.

Dashboard Page Navigation

You can select items in the views that refresh the page (drop-down menus or radio buttons) or click links to display pages with more detail. A list of links to previous pages (breadcrumbs) is located at the top of the page. The current page is highlighted in bold. At any time, you can move back using the breadcrumbs.

You navigate around vFoglight dashboards from summary dashboards down to detailed pages (drill-down) and back up the hierarchy. Navigation is controlled through menus, links, drop-down lists, and radio buttons.

Caution The pages display dynamic data. Therefore, you should use the links, and not the built-in navigation controls of your browser. Using the links refreshes the data on the page.

Data and Data Sources Pages

The *Data Sources* tab is only visible to users with administrative privileges in vFoglight. On the left side of the page is a navigation tree. Each parent node represents a group of vFoglight servers, by version. Individual servers are listed below the parent nodes.

The Data page displays the available data objects that have been instantiated on the currently running monitored system for which views have been created and for which root queries have been defined. See “[Queries](#)” on page 75 for more information about the query mechanism.

The Data Sources tab allows you to add or remove multiple databases. Using the query mechanism, Vizioncore Web components look for data in the data source to populate the views.

Definitions Pane

The *Definitions* pane is your workspace for creating, viewing and editing items in the *Web Component Framework*.

Note If a user is not assigned to the Cartridge Developer role in the Administration area of the browser interface, then he or she cannot see the Data or Definitions tab under Configure.

To open a definition:

Note Views may be opened for editing from the **Design** tab in the action panel.

- 1 From Configure, open **Dashboards > Configuration > Definitions** in the navigation panel.



- 2 You can use the Definitions pane to view or edit these components after you have selected them in the Module Contents pane. In the Definitions menu choice, the Module List pane displays the available modules, while the Module Contents pane displays the contents of the selected module. It contains a set of eight tabs, each listing a particular grouping, such as views, queries, or renderers.
- 3 In the Module Contents pane, open the item whose definition you want to view by selecting it. Alternatively you can create a new view or query. See [“Configuring Views”](#) on page 102 for information about creating a new view or see [“Context and the Context Tab”](#) on page 122 for information about creating a new query.

The various configuration options are:

| Data Display | Description |
|---------------------|--|
| View | Views are the <i>Web Component Framework</i> objects that display data. The components that are shown in the Module List pane are those that have been defined for the node you have selected in the Module Contents pane. See the <i>Web Component Reference</i> online help pages. |
| Queries | Queries let you select a set of data objects of the same type. Pages and views extract and use data from the queries for values and context settings. See Queries . |
| Renderers | A Renderer determines how a value is displayed. Various renderer types are available for customization. See Renderers . |
| Tasks | Tasks are applications external to the <i>Web Component Framework</i> that can be launched from within a view. See Tasks . |
| Icons | Each icon can be a collection of images of different size. See Icons . |
| Files | Upload files such as images to the module's public directory. See Files . |
| Types | Type mappings associate entities such as renderers, icons and flows to a specific data type or data type property. At runtime, the <i>Web Component Framework</i> will look up the entities mapped to these types and properties to use as defaults if more specific entities have not been specified. The name of the mapping is a combination of the datasource name and the data type name. See Types . |
| Units | Maps the unit of measurement to a datasource and a renderer. The name of the mapping is a combination of the datasource name and the unit name. See Units . |

- 4 When a definition opens for one of the items in the module list, you can view and reconfigure the settings.

- 5 Alternatively, you can create a new definition and configure it using the definition tabs. For more information see, “[Definitions Pane Settings Tabs](#)” on page 107.

Web Component Framework Workflow

Before you start working with the *Web Component Framework* you need to define what objectives you are trying to accomplish. For example, if you want to monitor a specific application server at certain intervals and send alerts to a specific email address, you need to establish choose the components necessary to create this type of workflow.

If you have already created a query that retrieves what you want from existing data, you can start creating views to display the data. If you do not have the proper data to work with, you need to create queries to return the data. For more information about queries, see “[Context and the Context Tab](#)” on page 122.

Additional Documentation

For more information about the *Web Component Framework* see the following documentation:

- *View Component Tutorial*—This document walks you through examples of how to configure some simple views.
- *Web Component Framework*—online help pages. These pages contain a complete list of properties for each item, object, or component in the framework.
- *vFoglight User Guide*—This document describes the default dashboards and how to create dashboards and reports from existing views. For more information about individual components, see the *Component Reference Guide* in the online help.
- *vFoglight Command-Line Reference Guide*—This document contains information on scripting vFoglight via DOS command or Unix shell scripts.

Customizing the UI Quickly

See the *vFoglight User Guide* for information on *Create dashboard*.

Finding Pages: Bookmarks

See the *vFoglight User Guide* for information on adding *Bookmarks*.

Queries

In vFoglight, queries are the preferred method for retrieving data from a data source. For example, you can set up a query to retrieve all agents running in a monitored environment whose host name matches the name of a particular *Host* object.

Note You can also retrieve data in a more limited way using a Data Runtime Value object .

The objects that appear in the Data menu choice under Configuration are those that are returned by root queries. A number of default Root Queries have been defined. These allow you to browse the common data objects. Using this approach you can build a custom query that returns objects of interest to you and drag them on to your custom dashboard to create pages that meet your specific needs.

Note You can build your own root queries. Simply enable the **Root Query** checkbox in the *New Query* definition editor.

This section discusses how to work with query definitions and provides explanations of the choices a user encounters.

This chapter contains the following sections:

| | |
|---|----|
| Overview of Query Definitions | 76 |
| Creating a Query in vFoglight | 76 |
| Parameters in Queries | 94 |
| Creating a Query | 94 |

Overview of Query Definitions

Queries let you select a set of data objects of the same type. Pages and views extract and use data from the queries for values and context settings.

Each query defines:

- The data-source type
- The specific data source
- The path of the query (the root location from which the query searches)
- A type of object to select

Optionally, a query can search down under the root, drilling down until it finds objects of the desired type. In addition, it can restrict the objects it finds using conditions (or restricts them to the first N objects), it can sort the objects, and it can aggregate the objects into maximums, minimums, averages, and other aggregates.

Queries retrieve a list of data objects of the same data type (such as Host or AppServer). For example, a query on the Host data type may return a list that contains all the available Host data objects from the data source.

Queries always return a list of data objects. Even when there is only one object found that satisfies the query, the result is a one-element list. A query that does not find any objects returns an empty list.

Just like Views, Relevant and Allowed Roles can be set on Query Definitions. These are taken into account when filtering Root Queries by role in the Data Browser, a portal or a report builder. For more information, see “[Roles in Query Definitions](#)” on page 78.

Creating a Query in vFoglight

You begin the process of creating a query by clicking the Add button on the Queries tab of the Module Contents pane. A New Query Dialog appears with these choices:

- Blank Query
- Copy of

- Derived from

Note For more information about these functions, see [“View Commands in the Definitions Area”](#) on page 105. For more information about configuring queries, see [“Query Definition Settings”](#) on page 77.

Query Definition Settings

This section provides the details of the definition settings for a query. Unlike views a query definition does not have separate tabs for different groups of settings. All the settings are on a single page.

Name

This is a required field. You cannot create a new query with the same name as another query in the same module. A warning icon (!) is displayed if this happens. Move your mouse over the icon and a tooltip displays the problem.

Root Query

Check this box to mark the query as a Root Query. Root queries (and their execution results) are listed in the *Data* browser and the *Data* tab of a Portal or a Report Builder. They provide a way to organize the data presented in those locations.

Relevant and Allowed Roles may be set on Query Definitions as well as on Views. These roles are taken into account when filtering Root Queries by role, as is done in the data browser, a portal or a report builder. For more information, see [“Relevant Roles and Allowed Roles”](#) on page 78.

Hide Root

The Hide Root checkbox is only enabled if Root Query is checked. If checked, when the results of the root query are shown in the Data Browser or the Data tab of a Portal or Report Builder, the first item in the results list will be hidden. If the query results list does not contain one item, then this option is ignored. This is useful when you know that the results of the query will always return one item and you want to save users having to expand the node corresponding to the one item in order to see its children. For example, if you have a query that returns the root of a data source, hiding that root is useful so that users can more easily get to the data that is really of interest.

Comments

This is an optional field intended for use by developers. It is automatically filled in if you are copying a query.

Context Help

You can use this field to tell end users about the purpose of this query.

Relevant Roles and Allowed Roles

Roles in View Definitions

By assigning user roles to views, each view can be marked as having zero or more Relevant Roles and zero or more Allowed Roles.

- Each user has Allowed Roles (the roles they are allowed to be in) and Relevant Roles (roles for which this component was designed to be useful). Relevant roles and Allowed roles contain the same list of items.
- If a view has no Relevant Roles marked, it is assumed to be relevant to all roles. If a view has no Allowed Roles marked, it is assumed to be allowed to be used by all roles.
- 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.
- 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 unless 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.

Each view can be marked as having zero or more Relevant Roles, and zero or more Allowed Roles. The roles set on a view are compared to the user roles when a page access is attempted. Users are assigned roles and these are matched against the Relevant Roles and the Allowed Roles for a view. This restriction does not apply if the user has the appropriate permission or if the view is in the user's personal module.

Roles in Query Definitions

Roles are taken into account when filtering Root Queries by role in the Data Browser, a portal, or a report builder.

Each query can be marked as having zero or more Relevant Roles, and zero or more Allowed Roles. This is related to user roles. Users are assigned roles and these are matched against the Relevant Roles and the Allowed Roles for a query.

If a query has no Allowed Roles marked, all roles are allowed to use this query. If some but not all Allowed Roles are set on a query, then only users for whom at least one of their roles is also an Allowed Role on the query can see or in any way interact with that query. This restriction does not apply if the user has `CHANGE_SYSTEM_MODULES` permission (as determined by the *AuthorizationService*) or if the query is in the user's personal module.

If a query has no Relevant Roles marked, it is assumed to be relevant to all roles. In some situations, the GUI may not show a query as a choice to a user if the user's roles do not match the Relevant Roles of the query. However, there is an option in those situations to see all queries that are allowed, regardless of Relevant Roles.

Module-Level Roles

Roles can be applied globally to all the views and queries in a module by editing its properties.

To edit module-level roles:

- 1 Select **Dashboards > Configuration > Definitions** in the navigation panel.
- 2 Click the triangular button at the right of a module name in the Module List pane.
- 3 In the drop-down list that appears, click **Edit**.
- 4 In the Edit Module dialog, select the appropriate **Relevant Role(s)** and **Allowed Role(s)**.

Data Source Type

Each query operates on a given type of data source. For example, vFoglight is a data source for a vFoglight server. The Data Source Type must be selected before you start editing. You cannot edit the Data Source Type while you are editing the query, as changing the type also changes other settings.

In vFoglight, there is only one Data Source Type. This field is displayed as text only for your information.

Data Source ID

The Data Source ID is selected from a drop-down list box that lists the instances of the DataSource Type. Selecting the (Default) item on the list applies this query to the default DataSource. This is the most common choice.

Required Parameters

This section of the query definition is used to specify any parameters that are required by the query and supplied at run time by Query Selection Runtime Values that use this query.

To add a Required Parameter:

- 1 Click the **+** button.

A Required Parameter row is displayed with the following fields:

- **Name** input field
- **Object Type** drop-down list
- **List** drop-down list

- 2 Enter the name as desired.

The page automatically refreshes after you leave the **Name** field.

- 3 Select the object type from the drop-down list. This is the type of value that the required parameter can have. For example, it can be a data type from the data source, such as Host, or a simple type like String.
- 4 You can select **True**, **False**, or **Unknown** from the **List** drop-down. These options indicate if the data in the parameter is a list, not a list, or might be a list. If the parameter is a list, it must be a list of objects of the selected **Object Type**.

The screenshot shows the 'Required Parameters' configuration window. On the left, there are settings for 'Object Type' (Host), 'Root Path' (Root /), 'Max Search Depth' (0), and 'Aggregations' (Filter results based on top N?). Below these are options for 'Value of N' (Specific Value, From Parameter) and 'Results to Include' (Only the 1, Everything). The 'Where' section is currently empty. On the right, a tree view shows the object hierarchy, with 'HostModel/hosts/monitoringAgent/alarms/sev...' selected. The tree includes nodes for 'monitoringAgent', 'serviceLevelPolicies', 'modelAlarmWarningCount', 'modelAlarmCriticalCount', 'modelAlarmFatalCount', 'modelChangeCountDelta', 'hosts', 'alarms', 'aggregateAlarms', 'aggregateAlarmState', 'changeSummary', 'monitoredHost', 'monitoringAgent', and 'alarms' with a sub-item 'severity'.

In the preceding example, the Hosts node is the name of the list of objects of the data type Host. A query always returns a list of data objects of the given type. Even if there is only one data object to return, the query returns a list with one item. A data object property can be a list property. For example, the Severity property in the Host data object is a list. Queries can also define and use parameters in the Root Path.

The screenshot shows a 'Required Parameters' dialog box with the following fields and values:

- Name:** MyHost
- Object Type:** Host
- List:** False
- Object Type:** AlarmSystemEvent
- Root Path:** Root / with Path {MyHost}
- Max Search Depth:** 0

In the preceding example, a Host data object has been added as a MyHost parameter to the Search Depth.

This tells how many levels below the specified Root Path the query is allowed to look for objects of the specified Object Type. The default is 0, and would apply to cases such as searching for objects of type Host under the root Hosts. Since Hosts is the list of all hosts, there is no need to search down further levels.

An example of using the value 1 would be if the root was Hosts and you wanted to find all events directly under those hosts by specifying an Object Type of Event.

Object Type

The Object Type is selected from a drop-down list of the available types for the query's Data Source Type, such as Host or AppServer.

Localized Names in Drop-Down Lists

Many of the fields are drop-down lists or drop-down trees. When these are opened, they display the localized names of the data objects or properties. Once you have made your selection, the drop-down closes, and the new name of the object is displayed. For example, the **with Path** drop-down tree shows the Java EE node. Expand this node and the App Servers node is displayed. When selected, the path displays as *javaee/appServers* in the field.

Root Path

This section of the definition sets the path to the Root, from which the query searches for objects of the given DataSource Type. The root must be either a data object or a list of data objects. The Root Path is not affected by the selected data object in the *Object Type* field, and shows the entire vFoglight schema.

Hide Root

The **Hide Root** checkbox is enabled only if **Root Query** is checked. If checked, when the results of the root query are shown in the Data Browser or the Data tab of a Portal or Report Builder, the first item in the results list is hidden. If the query results list does not contain one item, then this option is ignored. This is useful when you know that the results of the query will always return one item and you want to save users from having to expand the node corresponding to the sole item in order to see its children. For example, if you have a query that returns the root of a data source, hiding that root is useful so that users can more easily get to the data that is really of interest.

Aggregations

A query returns a list of data objects. Instead of directly returning these data objects, the **Aggregations** settings creates new data objects that contain only aggregated values. For example, if a query returns a set of alarms, you can replace these data objects with an Alarm data object containing the maximum severity of those alarms.

Note In most cases, the list contains only one aggregated data object.

Aggregation data objects are of the given data-object type, and they contain aggregated values of certain properties within the data objects. The aggregation types are:

- Maximum
- Minimum
- Sum
- Average
- Weighted Average
- Count

Unlike the other aggregation types, a count aggregation returns a Count data object with only two properties:

- Value - the count of returned data objects
- TypeName - the name of the type of data object being selected by the query and counted

Count is also different from the other aggregation types in how it handles a query that does not return any data objects. The other types will not create an aggregated object in this case. Count does create an aggregated object, with Value set to 0.

Creating an Aggregation

You can create more than one aggregation on the same query. If you do this, multiple aggregated data objects are returned in the results of the query: one for each aggregation, in the order specified. Leave the field blank if you do not require an aggregation on a property.

To add an aggregation:

- 1 Click the ⊕ button under **Aggregation**.
A new set of fields for an aggregation is added.
- 2 In the **Calculate** drop-down list select how you want the property field calculated.
- 3 In the **Property** drop-down list, select the type of property used in the aggregation.
- 4 Click the ⊕ button with the tooltip *Add Aggregation* if you need to define another aggregation.
- 5 Continue with “[Identifying Values](#)” on page 84.

If you aggregate on Metric properties, such as *cpuUsage* under Hosts, it creates a data object of the type Host, with a *cpuUsage* property, and that property has current, period and history properties. Each of those last three (the *MetricValues* within the Metric) have been aggregated, as follows:

- For a *max* aggregation, it aggregates the *max* property of the metric values, and stores it back into the *max* property in the aggregated *MetricValue*.
- For a *min* aggregation, it aggregates the *min* property of the metric values, and stores it back into the *min* property in the aggregated *MetricValue*.
- For an *average* aggregation, it aggregates the *average* property of the m metric values, and stores it back into the *average* property in the aggregated *MetricValue*.
- For a *weighted average* aggregation, it aggregates the *weighted average* property of the metric values, and stores it back into the *average* property in the aggregated *MetricValue*. The count property of a metric is used for the weight. For example, if the collected metric is Response Time, its average should be calculated by weighting the various observed values by how many times each different value occurred. On the other hand, if the system is monitoring two hosts, a simple average of CPU usage is appropriate to guard against the case where the sample rates for collecting data on the two hosts are different. If sample counts were used


to weight the average, the host that was sampled more frequently would skew the result.

- For a sum aggregation, it aggregates the average property of the metric values, and stores it back into the sum property in the aggregated *MetricValue*.
- The only situation in which each history object within the Metric is not aggregated is if the *timeRange* used for the Query uses RAW granularity, which means that each observation from the agent gets its own history row. In this case, the history lists from the various metrics being aggregated do not match up (in number, or in date/times), so there is no way they could be aggregated.
- The most likely properties that you will be interested in, when doing aggregations on Metric properties, are as follows:
 - For a max aggregation, the *period/max* under the Metric, which gives the maximum value for the whole *timeRange*.
 - For a *min* aggregation, the *period/min* under the Metric, which gives the minimum value for the whole *timeRange*.
 - For an average aggregation, the *period/average* under the Metric, which gives the average value for the whole *timeRange*.
 - For a sum aggregation, the *current/sum* under the Metric, which gives the sum of the values on the most current observations.

Note You are not limited to using these values if something else is desired.

Removing an Aggregation

To remove an aggregation:

- Click the  button at the right side of the aggregation type.

Identifying Values

The expected use case for Identifying Values is as a label for a query that returns an aggregated result.


Within each aggregation (except for ones using the operation count), you can also create Identifying Values for the object returned by the query. This is a way to assign fixed string values to other properties within the data object that is created as the result of choosing an aggregation. For instance, if you created an aggregation of events to get their maximum severity, you might add Identifying Values that returns just the Name

property of this object. The identifying value can then be used as a label for the row of a table that presents this information.

Adding an Identifying Value


Identifying Values settings only appear after you have created an aggregation.

To add an Identifying Value:

- 1 Click the  button under **Identifying Values**.
A new set of fields for an Identifying Value is added.
- 2 Select the **Property Name** (relative to the Object Type) and the **Text Value** to be assigned to that property.

Removing an Identifying Value:

To remove an Identifying Value:

- Click the  button located right of the Identifying Value.

Filter Results Based on Top N

The Filter Results Based on Top N property lets you restrict the results of a query. You can restrict the results to the first N or, conversely, exclude only the first N. For example, you may only want all but the first five items of the results from the query. This function is enabled by selecting the **Filter results based on top N?** checkbox.

The **Value of N** must be a number. You can set this using one of two radio buttons:

- **Specific Value**, where you enter a number greater than 0.
- **From Parameter**, where you can select a Required Parameter from the drop-down list and use the value.

Note You can only use a parameter if it returns a number.

The **Results to Include** lets you restrict the results using one of two radio buttons:

- **Only the Top N**, where only the first N results are returned.
- **Everything except the Top N**, where everything except the first N results are returned.

Selecting **Everything except the top N** may be useful in conjunction with an aggregation operation where you have used one query to get an average of the *top N*,

and you want to use another query to get an average of the rest of the data that are not in the *top N*.

You would almost always use Order By when you use Filter Results Based on Top N. Otherwise you would not know in which order the results would occur, and thus it would not be clear which results would be included in the *top N*.

Order By

Order By keys control the sorting of the data objects returned by the query.

Note Sorting is applied before Filter Results Based on Top N is applied.

You can create multiple keys. Order By sorts on the first key, then on the second key, and follows the sequence. Order By sorts only on a second key if the first key has data objects in which the values were identical. In this case Order By sorts the data objects containing those identical values based on the second key. This continues as further keys are specified.

There are three types of views that can use the query's sort order: drop-downs, row-oriented tables, and radio-buttons. As a result, sorting within a query is usually only needed when Filter Results Based on *Top N* is also used.

Note Row-oriented tables also have their own sorting capabilities. If used, these settings override the query's sort order.

If you are sorting on a metric property, such as *cpuUsage* under Hosts, the algorithm automatically sorts on the metric's current/value property.


Adding an Order By Key

To add an Order By key:

- 1 Click the **+** button under **Order By**.
A new set of input fields is added.
- 2 Select the **Path** from the drop-down tree.
- 3 Select **ascending** or **descending**.

Removing an Order By key

To remove an Order By key:

- Click the  button.


Where

The Where property restricts which data objects are selected by the query, based on criteria such as property values in the data objects.

Note In SQL, the property and its parts are also called Where clauses.

Adding a Where Clause

To add a Where clause:

- 1 Click the  button under **Where**.

A popup appears, displaying the following types of conditions:


- Comparison
- Is Set
- Sub Type is
- And
- Or
- Not

- 2 Select a type and the fields that define the condition are displayed.

Any condition evaluates to either true or false. The conditions in the Where clause are evaluated for each data object found by the query. That data object is included in the final results only if the condition evaluates to true.

Removing a Where Clause

To remove an entire where clause:

- Click the  button to the right of the section you want to remove.

Conditional Types

You can configure the following different types of conditions.

Comparison

This is the most common type of condition. A comparison consists of:

- A drop-down tree for the Path
- A drop-down list box for an operator
- A value that is used for comparison

The path on the left of the operator is compared to whatever is selected on the right of the operator, using the selected operator.

The operator can be:

- = (equals)
- != (not equals)
- < (less than)
- <= (less than or equal to)
- > (greater than)
- >= (greater than or equal to)
- like
- in

For the '=', '!=', and 'like' operators, the case insensitive attribute applies (the corresponding checkbox is enabled). Note that the case insensitive attribute will only be used if *both* the right hand and left hand values are strings.

The comparison value is set in one of two ways:

- If you select the **Specific Value** radio button, type the desired value into the input field.
- If you select the **From Parameter** radio button, you can select one of the Required Parameters (that have been defined in the query already) from the drop-down list.

You can use the value of the parameter. You can also drill down to a lower-level path in the parameter's type by selecting a node from the **Path** drop-down tree.

For example, if you are selecting Hosts, and want to only select one host named *MainServer*, you would enter the path of name, the operator =, and the specific value *MainServer*.

When the `in` operator is used, the value on the right must be a list. This can occur when parameters are used. The comparison is true for a given data object if the value in the **Path** field is contained in the list specified by the **From Parameter** field on the right.

The like operator is used for wildcard matching, and behaves exactly as it does in standard SQL. The pattern of characters and wildcards in the right field are compared against the entire value of the entry in the Path field.

The wildcards are:

- Underscore (`_`): represents any one character.
- Percent (`%`): represents any string of 0 or more characters.

A match is found only if the pattern is a complete match against the Path value. For example, the following patterns will all match MachineOne:

- `M%One`
- `%chi%`
- `Ma_hine_ne`

But `chi%` is not a match because it only matches a part of MachineOne.

Paths are allowed to have several levels, (such as events/name under Hosts), if an appropriate comparison can be made with the results.

The `<`, `<=`, `>` and `>=` operators can be used with numbers, strings or dates. If the Path on the left of the comparison evaluates to a number, and a Specific Value on the right of the comparison is a string, the string is treated as a number. If it cannot be converted to a number, the comparison will evaluate to false.

Is Set

Is Set requires a single Path selected from the drop-down tree of data objects. It evaluates to true if the value of that path is not empty (not null) and is not an empty list (in the case when the property always evaluates to a list).

In the rare case where the path points to a list of lists, the Is Set is true if at least one of the sub-lists is not empty. For example, if the elements the query is selecting are WebApplications, and you use *Is Set* with a path of `appServers/slowestRequests`, the condition is true for a given Web application if it contains at least one AppServer that contains at least one request in its slowest requests list.

Sub Type is

Sub Type requires a type to be selected from a drop-down list of types. The listed types are the sub types of the selected Object Type for the query. If there are no sub types of the Object Type, you receive an error message if you try to select this type of condition. It evaluates to true if the object is of the selected type (or one of its sub types). For instance, if you are selecting with an Object Type of Host, and you use a Sub Type is condition with type *WindowsHost*, only the Windows Hosts is selected.

Note For this usage, you could just have selected *WindowsHost* as the Object Type and embed this condition in a Not condition, to select all Hosts that were not Windows Hosts.

And

The And condition is true if all of the conditions below it are true, and is false if at least one of them is false. Use And to apply more than one condition (usually two or more).

To add a condition:

- 1 Select **And**, and click the ⊕ button below it.
- 2 Create the conditions as required.

Or

The Or condition is true if at least one of the conditions below it is true, and is false if all of them are false. Use Or to apply more than one condition (usually two or more).

To add a condition:

- 1 Select **Or**, and click the ⊕ button below it.
- 2 Create the conditions as required.

Not

Use Not to reverse the sense of a single condition. The Not condition is true if the condition below it is false, and false if the condition below it is true.

To add a condition:

- 1 Select **Not**, and click the ⊕ button below it.
- 2 Create the one condition as required.

Combining And, Or, and Not Conditions

By using nested combinations of And, Or and Not, you can create complex conditions, such as including a Host if:

- Its name is *MainServer* or *SecondServer*.
- There are no events under it.

The following example displays how the Where section of a query is set.

The screenshot shows the 'Where' section of a query builder. At the top, a dropdown menu is set to 'And'. Below it, there are three main clauses:

- An 'Or' clause containing two conditions:
 - 'Path name = Specific Value MainServer' (with a 'From Parameter' option)
 - 'Path name = Specific Value SecondServer' (with a 'From Parameter' option)
- A 'Not' clause containing one condition: 'Path is set events'.

The And clause contains the Or and the Not clauses. The Or clause itself contains the two comparison clauses, and the Not clause contains an Is Set clause.

Sequence of Evaluation

In a complex query, the selected data objects are subjected to the different parts of the query in the following order:

- 1 The data objects are restricted based on Where - if present.
- 2 They are then sorted based on Order By - if present.
- 3 Some are then selected based on Filter Results Based on Top N - if present.
- 4 The remaining data objects are aggregated (if aggregations are present).

Note For more information about Queries, see the Web Component Tutorial.

Creating New Queries

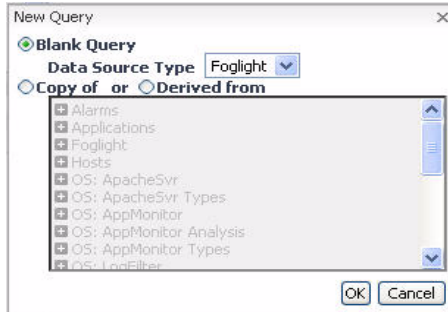
The New Query dialog lets you create a new query or a copy of an existing query.

To create a new query:

- 1 Select **Configuration > Definitions** in the navigation panel.

- 2 In the Module List pane, choose the module in which you want to work. If you are creating queries for your own use, choose **My Definitions**.
- 3 In the Module Contents pane, click the **Queries** tab.
- 4 Click **Add**.

The New Query dialog is displayed:



- 5 Select **Blank Query** and click **OK**.
- 6 A *New Query* tab opens in the View/Edit Definitions pane.

For queries, all settings are on this pane. There are no tabs as there are for more complex things, such as *Views*.

The *Definitions* pane changes to an *Edit* pane. It fills with the query editor, which contains all the fields that can be used to construct a query. You must fill in only the fields that are marked by an exclamation point icon on the right.

- 7 Type the name you have chosen for your query in the *Name* field.

Now that the query has been named, it can be referenced by that name when you want to use it.

- 8 Check **Root Query** if the query is to be used for this purpose.
- 9 If it is a root query and you want to hide the root, check **Hide Root**.
- 10 Check **Public** if you want to share the query.
- 11 Ensure that **Deprecated** is cleared. This checkbox is used to flag outdated queries.
- 12 Fill in the **Comments** and **Context Help** text boxes.
- 13 Select **Relevant** and **Allowed** roles for this query.
- 14 Select the **Data Source Type**. Typically, its value is **vFoglight**.

- 15 Ensure that the *Data Source ID* field's value is *<default>*.
- 16 Add any **Required Parameters** needed by the query.

These are values that are supplied at run time, usually from a context input.
- 17 Select the *Object Type* by clicking the drop-down arrow at the right of this field. Choose the object from the drop-down list.

A list of all the objects known to the running vFoglight instance appears. You'll be able to browse this object as soon as you have set the path in the next step.

Note This list box responds to keystrokes. You can type a letter and be taken right to the desired entry.
- 18 Searches normally start from the root (/), so ensure that *Root Path*, *Root* contains a slash (/).
- 19 Objects are grouped at some level down from the root. Choose the object you want from the drop-down list in the *with Path* field. You might need to set the *Max Search Depth* field as well.
- 20 You can have the query return aggregate values by setting *Aggregations* and *Identifying Values*.
- 21 You can filter results by specifying a value for *N*, either by giving it a specific value or by declaring a parameter that returns a value.
- 22 You can sort the returned result by choosing one or more items in a list.
- 23 You can use a *Where* clause to use a condition to limit the returned objects to the ones that match the criterion.
- 24 You can test the query by clicking the **Test** button at the top of the Edit pane.

The *Query Results* dialog appears, showing the name or names of the servers returned by the query and, in this case, the *Data Type* of the object, which is Host. The *Value* field is empty because there is no numerical value associated with objects. Simple types have values, objects do not.

Note You can see that the query did return a list of objects by clicking the expand icon (+) at the left any host name in the *Query Results* dialog. When you do, you'll see that some properties of the host, such as its *topologyObjectid* and *topologyObjectVersionId*, have numerical values, while others, such as *agents*, which are themselves objects, do not. The *Query Results* dialog allows you to see all the properties of the object, which is useful if you want to determine their names for use later on.

Note The *topologyObjectid* and *topologyObjectVersionId* properties mentioned in the previous note should not be used to select a particular object. If you do need to use a specific object in a custom view during a running session it is better to set an additional context value of type Data, type in a name for the Key, and navigate to the

particular runtime object in the Context Entry dialog. This is a much more efficient way of directly accessing the object rather than causing the system to search all objects until an ID match is found.

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

Parameters in Queries

Queries can be made aware of Query Selection Runtime Values. A Query Selection can have parameters. Their values replace the specified Required Parameters in the query. The Query Selection Runtime Value Parameters must exactly match the Required Parameters.

Currently, the query attributes that can be replaced by parameter values are the query's Root Path, the Filter Results Based on Top N value, and the right side of a comparison condition. The Root Path can only be replaced by a parameter that is a data object or a list of data objects. The value of the parameter becomes the root object for the query execution.

You can add additional path information under a parameter used for the Root Path. In this case, the parameter value is assumed to be a data object, but the value used is the given path into that data object.

For more information about using parameters, see [“Query Definition Settings”](#) on page 77 and [“Deriving a Query from Another Query”](#) on page 97. For information about aggregations see, [“Creating an Aggregation”](#) on page 83.

Creating a Query

Creating a query extracts data from a data source. Creating a query involves defining the following elements:

- The type of objects that you want to select.
- Any parameters that you want to pass to the query.
- The selection criteria, or the *where* clause.

If parameters are passed to a query, you can use them to compare the properties of selected objects to refine the *Where* clause.

To create a query:

- 1 In the Module Contents pane, click the **Queries** tab.
- 2 To create a new query, click **Add**.
The **New Query** dialog box appears.
- 3 To start from an empty query, ensure that the **Blank** checkbox is selected.
- 4 Select the data source type from the drop-down list.
- 5 To close the **New Query** dialog box and proceed to write the query, click **OK**.
The **New Query** dialog box closes, and the Definitions Pane appears in the pane to the right.
Note An exclamation mark that appears in the Definitions pane next to an input field indicates that the query cannot be saved unless you set that field. A corresponding tooltip also displays.
- 6 In the **Name** box type a name for the query.
When you are creating the query, save the query. That name appears in the **Queries** tab.
- 7 In the **Comments** box, type a brief description of the query you are writing. Alternatively, you can add more information using the **Context Help** box if desired.
- 8 Define the Data Source ID for the Data Source Type.
- 9 To add the parameter, under **Required Parameters**, click the plus icon.
A set of input fields appears. You can use them to further define the parameters that are passed to the query.
- 10 In the **Name** box, type the name of your choice—the one that you want to associate with the object type that is passed as a parameter to the query.
- 11 Define the object type by clicking **Object Type** and selecting a type from the list that appears.
- 12 Select one of the options in the **List** box to choose whether it is acceptable for the returned objects to be a list, not a list, or either.
- 13 Select an option from the **Object Type** drop-down list to define the type of objects that are selected by the query.
- 14 Ensure that the query searches for all *Object Type* instances in the collection model. To do that, verify that a forward slash '/' appears in the **Root** box, under **Root Path**. Then click the **Browse** button to the right of **with Path** and choose

the object that you want from the list that appears. Example:

The image shows a configuration dialog with two main sections. The first section is labeled 'Object Type' and contains a dropdown menu with 'PerformaSureAgent' selected. The second section is labeled 'Root Path' and contains a text input field with 'Root /' followed by a dropdown menu with 'with Path' selected, and another text input field with 'AgentMap/alarms' and a three-dot menu icon to its right.

- 15 To retrieve specific aggregated data, select values from the **Calculate** and **Property** drop-down lists.

Ensure that you have created an aggregation or that one is available.

- 16 To specify identifying values, click the ⊕.
- 17 Select values from the **Property** and **Text Value** drop-down lists.
- 18 To filter, check the **Filter Results Based on Top N**.

Continue to configure the filter options.

- 19 Under **Order By**, set the value and the order of the filter.
- 20 To add to a condition to the selection criteria, under **Where**, click ⊕.
- 21 From the list box that appears, select a condition type.

A set of input fields appears, letting you to further refine the output of the query.

- 22 Click **Save**.

Copying a Query

Copying is a fast way of creating a new query. It is also a way to create a modified version of a System query. You can copy any query, including your own User query, a System query, or a query created by another user.

You cannot create a copy of a query and give it the same name as another query in your module. If you enter a name that is already in use, an alert icon (⚠) is displayed beside the field and the **Save** button is disabled.

To copy a query:

There is no copy button for a query shown in the Definition pane. Instead, select the **New Query** button and the New Query dialog is displayed.

- 1 Select **Configuration > Definitions** in the navigation panel.
- 2 In the Module List pane, choose the module in which you want to work. If you are creating queries for your own use, choose **My Definitions**.
- 3 In the Module Contents pane, click the **Queries** tab.
- 4 Click **Add**.

The *New Query* dialog is displayed.

- 5 Select **Copy of** and from the drop-down list select a module that you want to copy.
- 6 Click **OK**.

The query definition is displayed.

Note To save this as a new query, you are only required to enter a name in the Name field. You can modify the other settings at a later time.

Deriving a Query from Another Query

Deriving a query from another query allows you to create a variant version of an existing query. You can derive from any query, including your own User query, a System query, or a query created by another user. You can even derive a query from another derived query. The query from which a derived query is made is called its base query, and is permanently linked to it.

You cannot create a derived query and give it the same name as another query in your module. If you enter a name that is already in use, an alert icon (⚠) is displayed beside the field and the Save button is disabled.

A derived query may extend its base query's list of Required Parameters, and replace its Filter Results Based on *TopN*, Aggregations, Order By and Where sections. If the derived query adds Required Parameters, their numerical order comes after that of the base query.

A simple example of where you might want to use derived queries is a base query that selects a group of Events, and a derived query that aggregates those events into a maximum severity.

In the editor, normally you are not allowed to edit or replace the Filter Results Based on Top N, Aggregation, Order By or Where settings in a derived query if those settings are already established in the base query. However, it is possible to get into a situation where both the base query and the derived query contain settings for any of these properties as follows:

- Start with a base query with the property, for example Aggregations, that is not set.
- Derive a query from it, and set Aggregations on the derived query.
- Go back to the base query, and set Aggregations on it.

In this situation, both queries have settings for Aggregations. The query editor will display warnings on both of these queries about that fact. If you edit the derived query again, you will be allowed to modify the Aggregations setting. However, if you remove that setting, it reverts to using the base query's Aggregations setting, and that setting is no longer editable.

Note Parts of the derived query that are inherited from the base query are displayed only in the Query Editor. They are not editable.

To derive a query:

- 1 From the vFoglight navigation panel under Dashboards, click **Configuration > Definitions**.
- 2 In the Module List pane, choose the module in which you want to work. If you are creating views for your own use, choose **My Definitions**.
- 3 In the Module Contents pane, click the **Queries** tab.
- 4 Click **Add**.
The New Query dialog is displayed.
- 5 Select **Derived from** and click **OK**.
The query definition is displayed.

Note To save this as a new query, you are only required to enter a name in the **Name** field. You can always modify the other settings at a later time.

Editing a Query

Follow this procedure to edit a query. Depending on your permissions, you may not be able to edit a System query or a query created by another user.

To edit a query:

- 1 In the Module Contents pane, click the **Queries** tab.
- 2 Select the User query from the Modules Contents pane.
- 3 Select **Edit** from the Definitions pane.
The query definition is displayed.

Deleting a Query

Follow this procedure to delete a query. You can only delete your own User queries. You cannot delete a System query or a query created by another user.

To delete a query:

- 1 In the Module Contents pane, click the **Queries** tab.
- 2 Select the User query from the List frame.
- 3 Select **Delete** from the Definitions pane.

A dialog is displayed asking you to confirm the deletion.

Configuring Views and Context

For a user interface component to be useful it must display relevant information. In most cases this information is not static, but is supplied by agents collecting data on an ongoing basis. Thus, there are two major requirements for a user interface framework—a rich set of visual components and a data binding mechanism. Queries are useful for retrieving data from a data source, but context is the usual mechanism for sharing the data among related views. This chapter discusses the tools for building visual components and describes how context is used to pass data among the various views that have a need for that data.

This chapter contains the following sections:

| | |
|---|-----|
| Configuring Views | 102 |
| Context and the Context Tab | 122 |

Configuring Views

If you have a data source you can start creating views. Views are collections of components that display data. The components of a view are self-contained, so you can add them to a page, remove them, or move them around according to your needs.

Containers hold the view components that present your data. A container view is comprised of multiple components that are organized by related content. Views are added to a container through the use of the Module Definitions editor. View components are the underlying configurable building blocks on which views are based. Each view component has both unique and common properties. For example, a table has a column property, a chart has a time-axis property, and both have a time-range property.

The view component groups available in vFoglight are summarized in this guide. For more information, see [“Anatomy of a Typical Dashboard”](#) on page 31.

Creating a New Container View

A container view is used to house data presentation views, such as tables, charts, and common UI components (for example, check boxes or labels). You can declare a container view to be a dashboard and use it to observe system performance at run-time. When creating a view, there are many to choose from. For more information, see [“Anatomy of a Typical Dashboard”](#) on page 31.

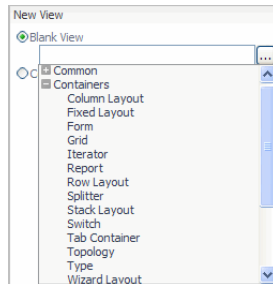
The choice of container depends on the components it is designed to contain and on the layout you want.

You can create a new container, but you are not able to add views unless you have defined them previously, so it is advisable to start by choosing the view components you need based on the data you want to present, and then decide on the container that is the best choice to present these views.

To create a container view:

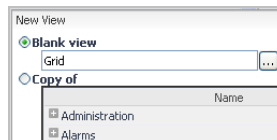
- 1 From the vFoglight navigation panel under Dashboards, click **Configuration > Definitions**.
- 2 In the Module List pane, choose the module in which you want to work. If you are creating views for your own use, choose **My Definitions**.
- 3 From the Module Contents pane, click **Add**.
- 4 In the **New View** dialog box, ensure that the **Blank view** radio button is selected.
- 5 Click the **Browse** button to the right of the **Blank view** box.

A list box appears, showing a tree of view types:



- 6 From the navigation tree, select **Containers** and click \oplus to expand the list.
- 7 Select the view.

The list box refreshes to show the newly-selected view type:



- 8 To close the **New View** dialog box, click the **OK** button.
- 9 The new, unnamed view appears in the editor pane.
- 10 Configure the container by filling in the required fields in the editor pane. The fields for any particular view are described in the *Web Component Reference* help pages and the *Web Component Tutorial* contains an introductory example of creating a container and various views for a dashboard.

Creating a New View Based on a Copy of a View

Copying is a fast way of creating a new view. It is also a way to create a modified version of a *System* view. You can copy any view, including your own *User* view, a *System* view, or a view created by another user.



You cannot create a copy of a view and give it the same name as another view in your module. If you enter a name that is already in use, a tooltip message is displayed beside the field and the **Save** button is disabled.

You can perform a normal or a deep copy of a view from the Definitions pane.

To create a new view based on a copy of an existing view in the Definitions pane:

- 1 From the vFoglight navigation panel under Dashboards, click **Configuration > Definitions**.
- 2 In the Module List pane, choose the module in which you want to work. If you are creating views for your own use, choose **My Definitions**.
- 3 Choose a view.
The view appears in the Definitions pane.
- 4 Click **Copy**.
A popup appears, with choices **Normal** and **Deep**.
Make your choice and a new popup appears, allowing you to select a target module.
- 5 Select a target module and the copy proceeds.
If any view is private, a popup alerts you and gives you the choice of copying these views as well.

To create a new view based on a copy of an existing view in the Edit pane:

- 1 From the vFoglight navigation panel under Dashboards, click **Configuration > Definitions**.
- 2 In the Module List pane, choose the module in which you want to work. If you are creating views for your own use, choose **My Definitions**.
- 3 From the Module Contents pane, click **Add**.
- 4 In the **New View** dialog box, ensure that the **Copy of** radio button is selected.
- 5 From the navigation tree, select a view type and click the  to expand the list.
- 6 Select the view.
The list box refreshes to show the newly-selected view type.
- 7 Click **OK**.
The Module Definitions pane switches to a new view in edit mode. The various fields contain the settings based on the copied component.
- 8 Make the changes you want, including renaming the component, and click  **Save**.
The component is saved in the module that was active when you clicked **Add**, which is normally your user space under vFoglight.

Definitions Page for a View


View Commands in the Definitions Area

There are seven commands for working with views in the Definitions area:

- Add (for creating new views only. This option appears in the Module Contents pane).
- Edit—switches the Definitions pane from view mode to edit mode.
- Remove—the component from the module.
- Copy—make a copy. See “[Deep Copying Views](#)” on page 106.
- Test—Verify the entity’s operation while it is under construction.
- Save—in edit mode, save the definitions for this component.
- Cancel—in edit mode, cancel editing and do not save changes.


You can find these functions in the Definitions pane in **Configuration > Definitions**.

Editing a View


Select the *User Definitions* area containing your view from the Module List pane, select the view you want to modify, then click **Edit** ( Edit) in the Definitions Pane toolbar.

Note For more information, see “[Definitions Pane Settings Tabs](#)” on page 107.

Saving a View

To save a view, the name field and required properties must be defined. Ensure that the view you want to save is in the **Definitions Pane**, and then select **Save** () from the **Definitions Pane** toolbar.

Removing a View

Ensure that the view you want to delete is in the **Definitions Pane**, and then select **Remove** () from the **Definitions Pane** toolbar. A dialog is displayed asking you to confirm the deletion. You can only delete views for which you have the proper permissions. You cannot delete a view if there are references to it.

Deep Copying Views

Ensure that the view you want to copy is in the **Definitions Pane**, and then select **Copy > Deep...** from the **Definitions Pane** toolbar.



Deep copying a view copies the selected view and all the entities and localized strings that are referenced using that view to the current module.

It takes into account:

- Context entries (view, flow, window) that are runtime values, which may have references to queries, renderers and localized strings.
- Configuration property values that are set to runtime values, which may have references to queries, renderers and localized strings.
- Configuration property values of type renderer, which when set have references to renderers.
- Flow, which may have references to tasks and to other views.
- Derived queries that are based on other queries.
- Container views that have references to other views.

Notes:

- Views containing non-public components are not supposed to be copied, but they can be deep copied. Also, you can perform a normal copy and choose which of the private views you require.
- The view to be deep copied must not be in the current module.
- All referenced views to be included in the deep copy must have their allowed roles satisfied by the roles of the current user.
- User modules do not have their own resource bundle, therefore if you deep copy a view into a user module, references to localized strings are not deep copied. The references are left as is and continue to refer to the original localized string.

Nested Views

These are private views that are contained only in a specific view. They cannot be added to containers other than the one in which they were nested.

When creating a nested view, the Context tab shows allows the user to define additional context for the nested component. The selected inputs are required in the nested view. All context inputs of the nested view are read-only.

A context input of a nested view might be converted to optional. This can happen if the nested view has an action configured that has generated context that is for input which causes an implicit (optional) context input to be created.

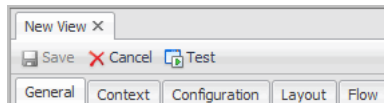
Replacing Views

Since system views are predefined views that are shipped with a WCF-based application, they may be referred to by other views so the IDs of these system views should not change. However, if you want to replace a system view with a new view that is based on a different component, you are not able to edit the view and change its component type. Instead, you create a new view and then replace the old view with a reference to the new view.

In order to do this, select the view to be replaced in the Definitions editor, then click the **Remove** button. Regardless of whether the view is removable (whether it has dependents or not), you have the option of replacing the view. If you choose to replace it, you then see a tree of views that includes only the views that have context inputs that are compatible with the required context inputs of the view to be replaced. Once the replace is confirmed, the view that was replaced no longer appear in the Definitions editor. Any references to it are automatically be converted to refer to the selected replacement view.

Definitions Pane Settings Tabs

If you have added a new view and are beginning to edit it, or when opening an existing view, the Definitions pane displays a series of tabs. These tabs enable you to configure the view definition properties. Each view is defined by a group of settings and each of these groupings has its own tab in a definition.



The type of View Component that you are creating determines which View Definition tabs display. See the *Web Component Framework* in the online help for vFoglight for the complete list of properties for each view.

Note The Views tab exists for some containers. The Layout tab is not available for some views.

Configuring Properties in the Definitions Pane

- [General Tab](#)
- [Context tab](#)
- [Additional Context](#)
- [Configuration Tab](#)
- [Views Tab](#) (displays for container views only)
- [Flow tab](#)
- [Layout Tab](#) (displays for some types of container views only)

General Tab

After you have created a new view, you need to configure its properties by completing the following fields.

| Field | Description |
|-----------------|--|
| Module | The module in which this component resides. |
| Component | The specific type of the view component, such as a Tree Table. |
| Name | The view name. |
| Public | Determines how this component can be referenced by other components. If true, the component can be used in any module. If false, the component can only be used in its own module or sub-modules of that module. |
| Deprecated | Removes the component from drop-down lists. |
| Preferred Width | The width of the view in pixels. |

| Field | Description |
|-------------------|--|
| Preferred Height | The height of the view in pixels. |
| Refresh Interval | The time interval after which the page refreshes itself by requesting new data from the server. |
| Priority | <p>The <i>Priority</i> setting is used to control which views are presented in the view pane of the Data tab when a node in the list pane is selected. The Data tab presents another way to browse vFoglight data. A running vFoglight system instantiates as many objects as it can from the data models defined in its agents and in its core system. These data objects are presented in the list pane.</p> <p>When one of these data objects is selected, the vFoglight system searches for any dashboard or monitor views that have as their sole context inputs a data object of this type. If a priority has been set for the view, it is displayed in the view pane. High priority views are shown first, and then those of lesser priority. If a priority has not been set, the view does not appear even though it has the proper input.</p> <p>Choices for the <i>Priority</i> setting are: None, Low, Medium, and High</p> |
| Purpose(s) | See “ Purpose of Views ” on page 110. |
| Custom Purpose(s) | See “ Purpose of Views ” on page 110. |
| Relevant Role(s) | See “ Roles ” on page 112. |
| Allowed Role(s) | See “ Roles ” on page 112. |
| Comments | A brief description about the view. |
| Context Help | A detailed description about the view. The description that you provide here becomes the online help for the component. When a user chooses this component from the Help drop down list, a popup containing the text is displayed. For this reason the text you provide here should fully describe the component from the user’s point of view. |

After the component has been saved the editor pane reverts to the definition pane. The following additional properties are presented:

| Field | Description |
|--------------------|--|
| Context Inputs | A table of context settings for the view. |
| Reference Id | The internal name for the component by which is referenced. Useful for exporting and importing custom views. |
| Last Modified Time | The time at which this view was last edited. |

Purpose of Views

Each view can be marked as having one or more purposes. The standard shipped purposes are:

- **Dashboard** — A dashboard is a page that does not require any input. The filtered list of views that you can add to a dashboard can contain any view that is purposed as a pagelet as long as the dashboard contains the appropriate context to make the view useful. Any view marked as a dashboard can be accessed in the *Configure > Dashboards* tab.
- **Diagnostic** — Useful for identifying views that are meant to be the beginning of a diagnostic work flow. This is the opposite of Monitor.
- **Dialog** — Useful for identifying views that are designed (in shape, size, and interactivity) to be Dialogs, either temporary or full.
- **Feed** — This view is useful as a syndication feed (for example, Really Simple Syndication (RSS) or Atom) used to frequently update content. Currently only a view based on Feed View Component should be purposed as a feed.
- **Global Action** — When set, a link to this view are shown in the right panel (action panel) under *Other Actions* as long as the user is permitted to see the view.
- **Home Page** — Useful for identifying views that are designed to be home pages.
- **Menu** — Useful for identifying views that are designed to be menus.
- **Monitor** — A view marked as a monitor is meant to reload itself periodically, since it is meant to monitor the data it is presenting. By default, the refresh interval is obtained from the user preferences but can be specified using the

refresh interval property of any view. The policy for refreshing views is as follows:

- Children views that are not marked as monitors refresh at the rate of their ancestor.
- Children views that are marked as monitors but are set to refresh at less than the rate of their ancestor still refresh at the rate of their ancestor.
- Children views that are marked as monitors whose refresh rate is more frequent than their ancestors refresh at their own rate.
- Children views marked as monitors but whose input *timeRange* do not have an idea of *now* (the current time) do not get refreshed. For a child page's *timeRange* to have a notion of *now* it must be passed a *timeRange*.

Note Note that monitor purpose affects the type of label that you see in the center of the page header as seen in the following policy code:

```
IF (there is a timeRange menu (topview has timeRange as a context
input))
    Show TimeRange
ELSE
    show no label
ELSE Show just time stamp
```

- Page — A page is a view that is designed to be used as a page – one that is drilled down to from another page or dashboard or portlet.
- Pagelet — A pagelet is a view that is designed to be added to a page or dashboard.
- Portlet — A portlet is a view that is designed to be added to a portal.
- Report — A report does not require any input. The filtered list of views that you can add to a report can contain any view that is purposed as a reportlet as long as the Report contains the appropriate context to make the view useful. Scheduled reports that have been generated appear in the Reports view.
- Reportlet — A reportlet is a view that is designed to be added to a report.
- Portal — A page that a user can customize by dragging views on to it. The filtered list of views that you can add to a portal can contain any view that is purposed as a portlet as long as the portal contains the appropriate context to make the view useful.

- **Summary** — Useful for identifying views that are designed to be small read-only summaries of data. Often useful for dwell actions.

In addition, you can enter custom purposes in the View Editor using a comma-separated list in a single text field.

Custom Purposes

You can enter custom purposes, using whatever names you want. Currently, in the View Editor, the custom purposes are entered as a comma-separated list in a single text field.

No actions are associated with custom purposes in this release. In future releases, custom purposes are used in various places in the GUI for restricting the list of views that can be seen to those who have been assigned the same custom purpose.

Roles

For more information, see [“Relevant Roles and Allowed Roles”](#) on page 78.

Comments

Creators of view components use this field to communicate information that might be helpful to other developers in creating similar components.

Context Help

Use this field for context sensitive help text. If the component is a top-level view, whatever you type here appears in the right action pane under the **Help** tab.

If the view is within a container and its border is showing, context sensitive help is available by clicking on the help icon in the title bar.

Context tab

Context packages the values that the page or view requires to present its data. For more information, see [“Context Types”](#) on page 124.

Pages and views are similar to forms waiting to be completed. Their definitions can specify the type of data that they display, just as a form can require a name or an address, which are filled in when the form is used. They do not set specific values for the item(s) that specify the subject of the form. For views, this is provided by the context data during run time, which is declared in the Context tab and is set in the

Additional Context group for the current view or page. If a context has been set by some view, it can be used in other views by declaring it in that view's context.

Context Inputs

If the context of a page has been set, then a view can use that value by specifying the named value of the context. For example, the *Host* page has set its current host into the context using the key `Host`. The *Event Summary for Host* view uses that context by setting its context input (the expected context value) to use `Host` as its key.

Similarly, if a page has a given named value in its context, and a link takes it to another page, then that new page can get access to that value by specifying its name as a context input.

Context can be passed when moving from one page to another. For example, click on a link in the *Host Monitor with Host Detail* view, and the *Host Browser* page is displayed with data for the same host. In this case, the *Host Browser* page expects that:

- a list of *Host* data objects is required (*hosts* context)
- a specific *Host* data object may be provided (internal context input)

The host is set in the Context Inputs section.

If a view is not passed the required context input, it uses the page's context.

Additional Context

Additional Context for a view allows you to set values on context keys for the view to use. If it is a container view, the additional information is used by its children.

Outputs (Global Context Mappings)

Some components may generate context values based on actions that the user performs on the component. These are listed in Global Generated Contexts in the reference page for the component. You can use this section of the Context editor to define a key for one or more of these context outputs if you need that information in a dependent view.

Configuration Tab

Configuration properties are those that control the overall look of the component and where the component gets its data. For instance, in a *Splitter* you can control whether the split is horizontal or vertical. In a simple component like a *Label*, two important configuration properties are the label text and background color. In a complex

component like a row-oriented table, there are a large number of properties for controlling how the rows are configured. In addition to the main property for setting values on rows, there are others for sorting, setting headers on columns, controlling width, and many others. For information on configuring the properties of individual components, see the *Web Component Reference* pages.

For an introductory overview of the configuration process, see the *Web Component Tutorial*.

Note You can use the Edit View Properties dialog to edit some of general properties of views without going to the Definitions pane. Click the **General** tab in the action panel. Under Actions, choose **Properties > Edit Basic Properties**. You can set relevant and allowed roles, the page refresh rate, and the context help text for the page.

Cooperative Layout

Some container views (currently: Row Layout, Column Layout, Fixed Layout, Grid, Iterator, and Tab Container) allow cooperative layout to be turned on via the *Cooperative Layout* property. When this feature is set, eligible child components (currently all chart components) of the same size align themselves within the container. In the case of the chart components, charts of the same width align their *y* axes to the same horizontal location within the charts and charts of the same height align their *x* axes to the same vertical location within the charts. Individual charts may be exempted from cooperative layout by setting the *Cooperative Layout* property on the chart to false.

By default most containers have cooperative layout set to **inherit**, which allows a parent container with cooperative layout enabled (if one exists) to include the subcontainer's children components in its cooperative layout. (The exception is the Tab Container, which defaults cooperative layout to disabled for performance reasons, and the Portlet container, which defaults to on.) Setting cooperative layout to **off** means that the container's child components will not be laid out cooperatively, no matter what the setting of its parent container is.

Flow tab

When you click a link on a page, what displays next depends on a common set of actions called *Application Flows*. These actions are set in the view editor's Flow tab.

Flows can be set on parts of the component, such as a table's rows or individual cells.

The resulting action depends on the current dynamic context. In the case of a row action, the view switches to a new page that shows the details for the object being used to populate the cells of the currently-selected row. These actions are set in views, but the targets may be popups, or other pages, or even other applications. The actions themselves rely on a context to give them information they need.

Configuring Flows for Views

The following sections provided details about how to configure flows for a view.

Flow Types

The flow types are:

- [Update](#)
- [Next Page](#)*
- [Popup](#)*
- [Previous](#)
- [Sequence](#)
- [Select Tagged View](#)*
- [Select Type Flow](#)*
- [Choose Value](#)
- [Choose Type](#)
- [Invoke Task](#)
- [External URL](#)
- [Show Help](#)
 - Note that for the flow types marked with an asterisk (*) a diagnostic workflow is available via a checkbox. If you mark a flow as diagnostic and if the time range is connected to “now” (real time), it will automatically remain unchanged so that the data and the data window you are looking at does not shift until you cancel this option by resetting the time range.

Update

Update refreshes the page with the new context. Where the context key is restricted using an internal input, the change is restricted to components that are children and grandchildren of the view marked with the internal context input.

This action refreshes the context for all the dependent views. For example, if you change the context on *view A*, then the data in *view B* is updated, and the data in *view C* is updated. By default, the context is initialized the first time a page or view is displayed. You may want to re-initialize context when more than two views are dependent on each other. If *view A* is not set to re-initialize its context, then *view B* updates, but *view C* does not change.

There is a *Preserve current timestamp* option, which if enabled allows an update to the page without changing the timestamp. This can improve response because fewer components need to be re-rendered.

Note A view whose context is marked as optional cannot be added to a container that has the same input, otherwise if a flow of type Update is triggered in this component, the context is retrieved from its container. The view's context entry won't replace the one in the container.

Next Page

Next Page sets specific target pages. Select the target from a tree of pages. The tree is filtered to only include non-deprecated pages, and dashboards (that have their required context inputs (if any) satisfied by the definition being edited). The purpose of the target page must be *page* or *dashboard*. This means that the type and the list property of the inputs must be satisfied. A target page must have its public attribute set unless it is in the same module hierarchy as the page that references it.

Popup

The tree of target views is filtered to only include non-deprecated views that have their required context inputs (if any) satisfied by the definition being edited. This means that the type and the list property of the inputs must be satisfied. Also, the list of options is filtered to match views whose purpose includes *pagelet*, *summary*, *menu* and *dialog*. A target page must have its public attribute set unless it is in the same module hierarchy as the page that references it.

A Popup has two different behaviors:

- Dwell – If the popup is associated with a dwell action then it shows the target view in a transitory read-only window. This disappears when the user clicks the mouse anywhere.
- Dialog – If the action is not a dwell then the target view is created in a popup dialog which the user can interact with, move about, and dismiss. This new popup view is part of the page and disappears along with the rest of the current page if you traverse to a different page.

Previous

When this action is invoked the page reverts to a previous state. There are two separate notions of the meaning of previous:

- If the view that initiates the reaction is somewhere on a page then the page switches to the previous one in memory. This is synonymous with clicking the previous page in the breadcrumbs trail at the top left of the page.
- If the view is a top level view (that is, it has not been drilled into or the only breadcrumb is the name of the view you are looking at) it has the same effect as using update on that page with the exception that the context is not updated.

If the root view causing the firing of the previous action is contained in a popup dialog then previous causes the dialog to pop down.

Sequence

The Sequence flow type is used when there is a requirement to do more than one type of flow, so it is used to execute a list of reactions sequentially. The order in the list is important and the execution of one reaction affects the next one, including changing the context.

This allows you, for example, to update the current page with a selection from a table (therefore causing the selected row to be remembered), and then go to another page. If you return to the first page, it is the updated page that is displayed. In this case the first reaction in the list should be Update and the second should be Next Page (reversing the order would cause the update to affect the next page, which is not the desired effect).

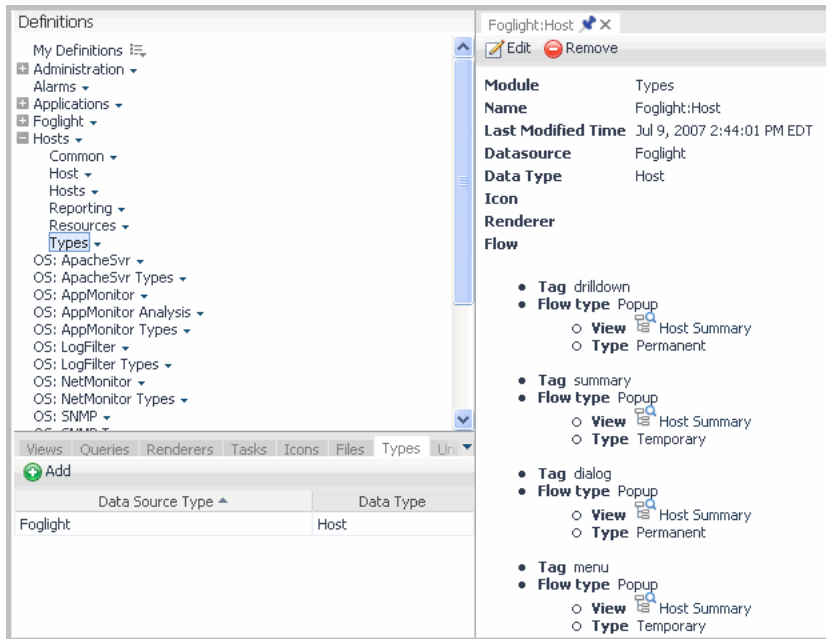
Select Tagged View

The Select tagged view reaction is used in vFoglight to implement Rule Based Drilldown. Its use is application specific, and is intended for internal development purposes.

Select Type Flow

A Type Flow is defined in a Type Mapping using the Types tab. You can associate a default UI flow (via a tag) with a specific data type. For example, you can define a default flow for a vFoglight Host to be a drill-down to the *Host Summary* page.

Type Flow links to selected pages when you specify a type of object. The figure shows a typical configuration. In this case, all cases flow to the same view, the *Host Summary* page. You have the option of marking the popup as temporary or permanent, which is the reason for having two drill downs to the same page. Also, the flow might be based on a menu choice.



To enable this Type Flow in a view, use Select Type Flow when setting up the flow type. For example, a cell selection on the Host column triggers a Type Flow based on the data type of vFoglight: Host, the host that is currently selected. If the preceding Type Flow configuration is in place and the selected item is a Host instance, the system drills down to the *Host Summary* page.

If *Select Type Flow* is the flow type, the vFoglight looks for all Type Mappings available in the system for a match. If an exact match is not available, the search continues up the data type inheritance tree until a match is found or reports an error if it fails. For example, if a *WebLogicServer* Type Flow is not defined, the system follows its inheritance tree to look for a match: *WebLogicServer* > *JavaEEAppServer* > *JavaEECollectionModelInstance* > *CollectionModelInstance* > *ModelInstance* > *TopologyObject*. If the search reaches the root without finding a match it reports an error.

The tagged flow mechanism allows more than one flow to be configured for each data type. Using tags you can specify a default dwell, a default temporary popup and a default drill down for each data type.

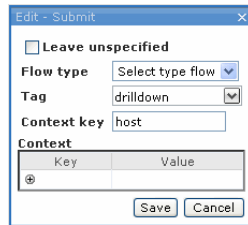
To define the flow for a view:

- 1 Open the **Flow** tab.

- 2 Click one of the actions.

The Edit window opens and displays a set of input fields that allow you to further define the flow.

- 3 Clear the **Leave unspecified** checkbox.



The screenshot shows a dialog box titled "Edit - Submit". It contains the following elements:

- A checkbox labeled "Leave unspecified" which is currently unchecked.
- A "Flow type" dropdown menu set to "Select type flow".
- A "Tag" dropdown menu set to "drilldown".
- A "Context key" text input field containing the text "host".
- A "Context" section with a table that has two columns: "Key" and "Value". The table is currently empty.
- At the bottom right, there are "Save" and "Cancel" buttons.

- 4 Click **Flow Type** and select one of the options.
- 5 If the type is Select type flow, choose one of the tags from the drop down list or type in a tag name of your choice.
- 6 Click the **Browse** button to the right of the **Context Key** box and choose a name from the list that appears.
- 7 Click **Save**.

Choose Value

Choose Value lets you select different actions and targets based on case values. For instance, you might want to have a different reaction depending on which cell of a table the user chooses. You would set a Choose Value flow type on the Cell Selection flow and then you would add cases for the various columns. You choose a name key for the column and associate a flow with it. This key name must be the same one that you assigned to the column ID under Column in the table's configuration tab. Thus a different reaction can be associated with each column in the table.


Follow the procedure given below to configure a case.

To set the Choose action:

- 1 Select the **Choose Value** action from the *Flow type* drop-down list.
- 2 Select a key from the *Context key* drop-down list. This is the key that you have configured to have different runtime values that are going to be used as case values.
- 3 Click **Save**.

The context is added to the action in the tree.

To add a case:

- 1 Click , whose tooltip is *Click to add a case*.
A new row for the case is added under the default row.
- 2 Click the new row.
- 3 Type a context key in the *Case Value* text field and specify a Flow type in the drop-down list.
- 4 Enter the context key for the case by selecting the field and then entering the key.
- 5 Select the flow type and add context entries if needed.
- 6 Click **OK**.

To remove a case

- Click  on the row of the case.

There is also a *Context* group box where you can specify any additional context to be passed to the target view. This may be necessary if the target view has a required context that is not already defined in the calling view's context.

Choose Type

Choose Type is similar to Choose Value, but instead of directing actions based on the value of a context element you can do this by the Type of the object. Often the reason to do this is that you have a generic object and depending on the more specific nature of the run time type you want to perform a different action.

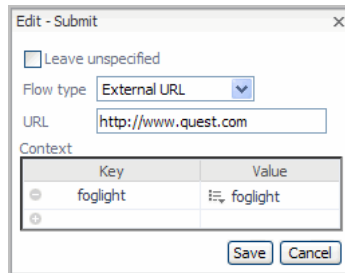
For example, use *Choose Type* if your type is Host and you want to go to different pages depending on whether the actual object is a WindowsHost or an AIXHost.

Invoke Task

Invoke task allows you to invoke a task when an action occurs. For example, you acknowledge an alert or launch a program on your desktop with information from the vFoglight application. The tree of tasks is filtered to only include non-deprecated tasks that have their required context inputs (if any) satisfied by the definition being edited. This means that the type and the list property of the inputs of the task to be invoked must be satisfied.

External URL

Instead of flowing to another vFoglight view, any Web page can be referenced by specifying its URL.



External URL lets you set any URL address as the link target. If you set the external URL to the string {url} the system looks in the context for a key URL and uses its value as the URL to go to. This is useful in conjunction with a template allowing you to customize the URL with runtime data from vFoglight.

Show Help

This choice is used to launch the help associated with a specific help key. This choice is useful for connecting alarm states or in deciding what to do with a particular section of the help. Note that you can connect a particular view, as long as it is a page or has a visible title bar, to the JavaHelp mechanism via a context ID.

Layout Tab

The Layout tab is used in the Fixed Layout and Grid containers. The tab contains a layout frame. At the top of the frame is a toolbar with these functions:

- Add—an existing component to the layout.
- Define Customizer—for placing a link to a view in the General tab of the action panel.
- Properties—the placement properties dialog for the selected component.
- Inspect—the chosen component by opening it in the Definitions pane.
- Remove—the selected component from the layout.

The Layout frame shows the views that are on the container and their approximate sizes. The guide lines in the background show you how the views would fit in browser windows with different screen resolutions.

Views Tab

The Views tab is used in the Iterator (where it is called an Iterator View), Report, Switch, Tab Container, Topology, and Type containers. The tab incorporates an editor for setting views for the container.

Context and the Context Tab

Context is the term used for the information available at a given point in time to the various components used by the *Web Component Framework*. Context in the *Web Component Framework* is similar to a collection of variables that are used in programming languages or an operating system's environment variables. Every element of the context has a key (name), a corresponding type, and a run time value in an active vFoglight instance.

Context consists of:

| | |
|-------|---|
| key | The strongly typed, case insensitive string that references a particular context element. Type checking is done at design time. |
| value | The value stored in the context element, which can be either a Web Component Framework data object or a Java object. |

For example, a view that is configured to display alarm data for any one host must be provided with a specific host to give it context. Similarly, a Memory Usage page, which displays memory data for a specific host given a time range, must be provided with two context inputs. In the first example, the host is the context. In the second example the host and the time range together comprise the context.

Pages and views can be compared to forms whose fields are filled in at run time. The context definitions determine the actual data that is specified by the Context, which is set in the following ways:

- A time range, which is available to all components.
- The Context tab in the View editor—you declare context inputs to the component here. You can define additional context entries here as well.
- The Configuration tab in the View editor—for example, by setting row/cell properties and choosing an available built-in context such as `<currentRow>`.

- The Configuration tab in the View editor—by setting a parameter’s value in the *Edit - Value* dialog for a component.
- In the Flow tab of the View editor—you can set additional context here, or map from internal component sources into the context.

For example, a page is designed to display information for a selected host. When you click a link, the host object is passed to a detail page (in this case, you have already set up the detail page to require host information when it is called). The detail page uses that information as its initial context settings. The Context tab is where you set up the list of required context data. Using a required Context setting means that the page or view is dependent for its data on that context value being available.

Context Tab

Context allows you to define the list of information that is required for a particular view to display the data that the page is designed to show. When a context element is defined it is given a name key and it is assigned a data type.

Global Context

In any application there is global or application level context, which is the information that is available throughout the application. An example of this type of context is *timeRange*. Each application can decide to make arbitrary information available to all views.

Dashboard Context

Context is implemented so that changes to one context element affect other elements in a hierarchy. If a context element changes in a nested level, the change traverses up the hierarchy to the parent level, assuming that an uninterrupted chain of key definitions exists.

User Context

When a user interacts with views the interaction can cause changes to the context. For example, when selecting a particular item in a drop down list, by default the *Web Component Framework* saves the last known values so that when you return to the page, it displays with that same item selected.

Note You can disable this behavior in User Preferences.

Context Types

See also “[Context tab](#)” on page 112.

There are three types of context entries:

- Required
- Optional
- Internal

If a view has only one input other than *timeRange* or only one required input (and zero or more optional inputs) it becomes its primary input. The primary input and its priority defines, whether it appears in data browsers when a node of that type is selected.

Required Context

If the input is marked as required, then it must be provided for the view to operate. You cannot add a component to a parent unless the parent can provide the required context. Thus, the parent view must do one of the following:

- Itself have that context as a required input.
- Specify that context by defining it as additional context.
- Specify that element in an instance context of one of its child components.
- Specify the element as an optional or internal input and contain another child that promises to provide a value for it.

Optional Context

If the input is marked as optional, then the view takes the data if present, otherwise it has some mechanism for inferring or calculating the data. Often this comes in the form of a drop down list or selectable table that auto-selects the first element in the data list if a selected item is not provided. Optional Context items are those inputs that are used if they are present but are derived in some other way if they are not present. For example:

- A user can define a default value for the input. This default value can take the form of a query selection or some other mechanism. If the optional input is not available then the default value is taken.
- Some components such as the Drop-Down List component and Row-Oriented Table can provide a value if one is not there. For example, if you have a drop down component with a list of Host objects you can automatically select one and post it into a Host key using the Flow Context.

Internal Context

Marking a context as Internal means that the input is required for internal operation of the table and is much like the optional input, except that changes to this value by the component are NOT propagated to the rest of the page on an update. However, the value IS propagated to the children of the component.

Internal Context items behave like optional inputs, except that when a context element changes in a nested level, the changes do not traverse up the hierarchy to the parent level. This makes it possible to have similar and dynamic views that use the same key names and types co-existing on the same page without adversely affecting each other.

You can use this type of context in a portlet with a customizer. For example, the customizer allows you to select which Host is displayed by the portlet. It is important for the component to mark the Host as internal to make sure that if two identical views are on the page at the same time, but each show data from a different host, that they do not conflict with each other.

Caution A view with an internal context cannot be added to a container that has the same context input, because if the view's context changes and the page is updated, this update are not be passed on to the container.
Internal Context is not for propagating context from one view to another within a page. It is designed to stop propagation of context when different parts of the page have different values for the same context.

For more information see “[Configuring Views](#)” on page 102.

Additional Context

Additional Context is context that is added by the view designer. You can add context using the Additional Context tab in the Module Contents pane to the following:

- Views
- Specific Instance of a View
- Flows

Additional Context for a View

Additional Context for a view allows you to supply additional information that your view can use inside its configuration or additionally, if it is a container view, for use by its children.

If you know that you are going to refer to a particular child of an existing context key often, you can create a new key that short cuts the reference. For example:

`agent = <host>/agent`, where `host` is an object that has a property called `agent`.

This is also useful if you are building a container view and want to include children that require particular keys that are not explicitly present.

Additional Context for an Instance

When adding a view to a container view, it sometimes happens that the available context does not have the right information for the view that you want to include. In this case, it is possible in the container view to selectively create the right context. This is called *additional context for an instance* or *instance context*. The context is for that particular instance of the view and is not available when you use the view somewhere else.

Note You would create the *additional context of the view* itself if you wanted it available to every instance of the view.

Additional Context for a Flow

The progression from one view to another is called a flow. How the new view is made to appear depends on the flow action. The types of flow actions are described in “[Flow Types](#)” on page 115.

Often these views may have input requirements that exceed what is available in the context of the component from which you are coming. In this case, it is possible to define further *Additional Context* that applies to the flow.

Dynamic Context

Dynamic context is only available for some components, such as Bar or Pie Chart, Cluster Bar Chart, Time Bar Chart, Time Plot Chart, Drop Down List, Filter, Radio Button List, Row-Oriented Table, Tree, Tree Table, and Topology. The value of a dynamic context item changes depending on what is being rendered at the time, such as a row in a Row-Oriented table, or what is being selected, such as an item in a list. For example, in the Row-Oriented Table the dynamic context key *current Row* is available. When a row is processed for display, that row’s *current Row* value changes depending on which row is currently being rendered.

Examples of dynamic context keys are:

| Dynamic Context Key | In Component |
|---------------------|------------------------|
| metric | Bar Gauge |
| currentDataParent | Cluster Bar Chart |
| currentItem | Drop-Down List, Filter |
| currentRow | Row-Oriented Table |
| currentNode | Tree |

There are other components that have properties associated with dynamic context keys. See the *Web Component Reference* pages for details.

Flow Context

You can create context keys that relate to a flow action. For example, suppose you want to select a host from a table showing a list of Hosts. To get access to the selected Host you have to define a context element for use as a Flow Context for table rows. To do this, define the *Selected Row* built-in context as *Host*. Then you can use it to change pages to a view that requires *Host* as an input. See “[Additional Context](#)” on page 125 to learn about adding more context elements to the flow.

Validation of Context in the Editors

When checking the required inputs of a contained view, optional and internal inputs are provided at runtime by either the container or another contained view.

Type Casting Context

The strict typing nature of the *Web Component Framework* disallows arbitrary casting of one type to another, since in any responsible UI development you need to detect and prevent incorrect input.

On-null

Many places in the flow and context definition allow you to specify the value to use if the incoming value is an unexpected null value.

TimeRange

Some special behaviors of the context element *timeRange* are:

- It is always available; even if it not is explicitly specified, the low level data access calls use its current application-level value.
- Explicitly including *timeRange* as a required Context on a view that becomes a page causes a time range drop-down to appear at the top right of the page. This enables the user to change the time range used in the page.
- Marking a *timeRange* as an optional Context in a component permits it to be independently updated from the rest of the views on a page. This permits multiple views on a page to each have their own time range.

Tip If you don't want the Time Range drop-down to appear on a page, remove *timeRange* from the context.

Runtime Values

This section describes the configurable properties for pages and views. Configurable properties determine which data and the format for the data items actually display in a View. Properties set at design time. They are typically simple types, such as numerical and string values. The view's Context contains its dynamic configurable variables. The properties for each View are set in its Configuration tab. Examples are a parameterized title, formatting, and the source of its data values or metrics.

This chapter contains the following sections:

[Configurable Properties and Runtime Values](#).....130

Configurable Properties and Runtime Values

Each configurable property has a specified data type. There are simple types and Runtime Value types. Simple types are types such as strings or numbers. For example, the font size property is a number. Simple types are typically for formatting or labeling properties.

The Runtime Value types are used to retrieve the dynamic data in a view. The values of these properties are extracted from data objects at run time. For example, all the monitored hosts can be extracted from the Host data object.

Simple Types

These types are set using the property editors available in the Configuration tab in Definitions.

The following table lists the simple types that are used to configure some properties:

| Type | Description |
|------------------|---|
| Boolean | A fixed value of true or false. The Selectable property uses this type. |
| Color | An RGB color specification. A color coordinate can be entered or a color can be chosen from the color palette. |
| Data Object Type | The type of a database object. |
| Enumerated value | A selection from a fixed list of options. Properties that use this type include: <ul style="list-style-type: none">• Header Alignment (options are <i>Vertical</i> and <i>Horizontal</i>)• Sort Order (options are <i>asc</i> or <i>desc</i>) |
| Error Renderer | The name of an error renderer, which displays a value indicating <i>error</i> if the attempt to evaluate in the associated bound data results in an error. See “ Renderers ” on page 148. |
| ImageReference | Contains a reference to an image and to its size. The reference can be indexed or non-indexed. |

| Type | Description |
|---------------|---|
| Null Renderer | The name of a null renderer, which displays a value indicating <i>no data</i> if the values in the associated bound data are null. See “Renderers” on page 148. |
| Number value | Any fixed numeric value, such as 123.45. Commonly used for integer values. Properties that use this type include: <ul style="list-style-type: none"> • Location • Size • Column or Cell Width (in pixels) |
| String | A fixed text string, such as <i>My Title</i> . Commonly used for IDs, which are used for flows, filtering and sorting. It can also be used for simple text labels. Properties that use this type include: <ul style="list-style-type: none"> • Font sizes (The String Template Runtime Value type is similar, except it accepts parameters that can be set at run time.) |
| TimeRange | A time range has a start and end time. Predefined time ranges may be tied to the calendar or the current time. Custom time ranges are open without restrictions. |

Runtime Value Types

There are several types of Runtime Values. Each one provides a way to define a value, which, depending on the type and the way it is used, may or may not be bound to data from the underlying system. A Runtime Value can evaluate to simple objects such as strings, complex objects, lists of objects, or lists of lists of objects.

The following table lists the available types of Runtime Values. The concepts of parameters and context, which are referred to in some of the descriptions, are explained in [“Parameters in Queries”](#) on page 94 and [“Context and the Context Tab”](#) on page 122.

| Type | Description |
|-------------------|---|
| Context Selection | Extracts a value from the Context, and can select parts of the context using a path (see For more information, see “Context Selection” on page 133.). |

| Type | Description |
|--------------------|---|
| Data | Specifies one bound data value or a list of data values directly, without using a Query. It is always bound to a specific data source. Because it is bound to a specific data instance, it can be used to eliminate an expensive query search (see “ Data ” on page 141). |
| Date | A Date object. (“ Date ” on page 142) |
| Icon Selection | Select an icon for display. If no Icon Renderer is specified by the user in the Icon Selection, a default Icon Renderer is used to display the icon at Normal size (see “ Icon Selection ” on page 138). |
| Image Reference | A reference to an image. (See “ Image Reference ” on page 140) |
| List | Allows the creation of lists of values either from individual values, or by merging existing lists. (See “ List ” on page 142) |
| Query Selection | Calls a named Query to retrieve bound data, and can select parts of the Query's results using a path and other arguments (see “ Query Selection ” on page 135). |
| Rich Text Template | Rich Text Template is similar to a String Template, but permits a restricted set of user-entered XHTML. It allows parameters to be set, while Rich Text does not (see “ Rich Text and Rich Text Template ” on page 140). |
| String | A selection from a list of localized string values, which can differ based on the language settings in the client browser. This runtime value is available only for components in system modules. |
| String Template | Specifies a simple static text string (such as a String value), but can also perform more complex tasks using parameters (see “ String Template ” on page 139). |
| Theme Selection | Selects a theme based on a Component, Style and Value. For more information, see “ Theme Selection ” on page 137. |

Details of each Runtime Value

The following types of Runtime Value are available to generate data for views:

- [Context Selection](#)
- [Data](#)
- [Icon Selection](#)
- [List](#)
- [Localized String](#)
- [Query Selection](#)
- [Rich Text and Rich Text Template](#)
- [String Template](#)
- [Theme Selection](#)
- [Return Types](#)

Context Selection

Context Selection Runtime Values in a configuration can access any value in the context.

Note Context Selection cannot have parameters.

The following table describes the properties of a Context Selection:

| Property | Description |
|-----------------------|---|
| Show Advance Checkbox | When enabled, the properties marked with a * are visible. |
| Input Key | The name of the item in the context that is to be selected. When context objects are defined, a name key is part of the definition. Access to the object is managed through its name key. |
| * Treat as Type | You can choose to specify one of the allowable sub-types of the object's base type. For example, if the context selection is a host, you might choose to restrict it to an AIX_Host. |

| Property | Description |
|-----------------------------|---|
| Path | The location of the desired property within the data object hierarchy. This displays the actual names of the data-object properties, and not the localized names. |
| Return First Object in List | <p>If you check Return First Object in List, the first element of the first list that is encountered is set to be a single item. For example, if the context element itself is a list then it uses the first element of the list, instead of the whole list. The optional Path is then applied to that first element.</p> <p>If, on the other hand, the last element is a list and there are no other lists in the path before it, then the first element of that last list is returned.</p> <p>If there are two or more lists in the path, only the first one is reduced to its first element. To reduce a number of lists to a single object it is currently necessary to build it up with multiple keys in “Additional Context” on page 125.</p> |
| Renderer | See “ Renderers ” on page 148. |
| * Return Type | <p>Context Selection can return ten possible types of data:</p> <ul style="list-style-type: none"> • Localized Value • Value • CountPerItem • Count All • Unit • Units • Localized Property Name • Localized Property Names • Localized Type Name • Localized Type Names <p>For more information, see Return Types.</p> |
| * Unit Property Name | If the Return Type is Unit or Units, then this selects and returns the specified property out of the unit, rather than returning the entire unit object. If the Return Type is neither Unit nor Units, this is not applicable. |

| Property | Description |
|---------------------|---|
| * Time Range To Use | You can choose the default time range, or all time, or any other time range that has been defined. |
| Parameter block | Allows you to choose a Runtime Value for On Null, or any other parameter that has been declared in the query. |

Query Selection

A Query Selection Runtime Value evaluates to a specified value in the data object or list of data objects returned by a query. For example, you can display a list of hosts from a selected data object.

A Query Selection has the following properties:

| Property | Description |
|-----------------------------|---|
| Query | The query that is run to determine the set of values to be used. Open the drop-down menu and select from a tree of queries. |
| Path | A path within the results from the query. This displays the actual names of the data-object properties, and not the localized names. For more information, see the Data Type Reference. |
| Iterate Over 1st Parameter | For details, see “ Configuring the Query Selection ” on page 136. |
| Time Range To Use | If this is checked, the TimeRange currently in force is ignored, and data for <i>all time</i> is selected. |
| Return First Object in List | This lets you select the first item out of the results of the Query property, which is always a list. The optional Path is then applied to that specified element. |
| Renderer | See “ Renderers ” on page 148. |
| Parameter block | Allows you to choose a Runtime Value for On Null, or any other parameter that has been declared in the query. |

Unlike String parameters, Query Selection parameters are not used directly by the Query Selection. Rather, they are evaluated by the Query Selection, and then passed on

to the underlying query, which is executed by the Query Selection. For more information, see “[Parameters in Queries](#)” on page 94.

Parameters in a Query Selection are often used to specify the root path of a query. If the root is not a parameter, it is an absolute path from the root of the data source. If a query's root is specified with a parameter reference, then the corresponding parameter from the Query Selection is assumed to have evaluated to a data object or a list of data objects. That value is used as the root object for the query.

The behavior becomes more complex if the root is a parameter that evaluates to a list of data objects, and the query specifies aggregation, such as *Max*. In this case you have a choice:

- a Obtain just one aggregate value for the result of the query against all of the elements in the list of data objects that is the value of the root object's parameter.
- b Obtain one aggregate value for the result of the query against each element in the list that is the value of the root object's parameter. The result is a list of aggregated values.

Note For more information about aggregation, see “[Aggregations](#)” on page 82.

For example, you have a view which is a row-oriented table that displays a list of hosts down the rows. You want to see the maximum severity of events for each host in another column of the table, which requires using option *b*. If the root parameter is the list of hosts, and the aggregate Query is selecting the maximum severity of events, then the results are a list of the maximum severities of the events for each of the hosts (one maximum per host).

Configuring the Query Selection

To configure the query selection:

- 1 Select the **Values** property of the desired column of the row-oriented table view.
- 2 Set that property to be a **Query Selection** accessing the correct query.
- 3 Check the **Iterate Over 1st Parameter** in the Query Selection.

Caution Failure to follow these steps results in the query showing just one value for all the elements in the list (option *a*).

If Iterate Over 1st Parameter is selected, then the query executes once for each row. This mechanism is available only for the first parameter in a query with multiple parameters. It executes for each element from the list that is the value of the first parameter. The results of each of those query executions are amalgamated into a list, which becomes the final value of the Query Selection.

If a Query Selection uses Iterate Over 1st Parameter and has an On Null Runtime Value, the On Null may need to access the current value being iterated over during the Query Selection evaluation of the list that is set as the first parameter. To make this possible, the current value from the first parameter is put into the context with the key *currentParameter* prior to evaluating the On Null.

Caution Never create a user-defined context key with the name *currentParameter*.

Queries always return a list of results even if there is just one element in the list, as is the case with aggregation queries. Therefore, if a Query Selection uses Iterate Over 1st Parameter and has a list of hosts as the parameter being iterated over, then the result returned is a list containing one result element for the evaluation of the query for each host. In most cases, each result element is a one-element list containing the result data object.

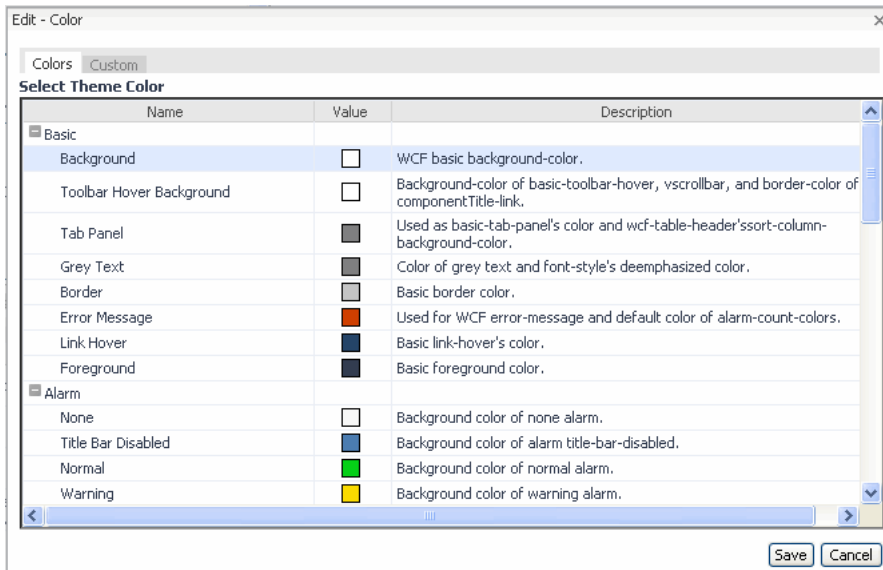
To easily access this sole result element, check Return First Object in List. This selects the first element out of each list of query results. When Return First Object in List is used in combination with Iterate Over 1st Parameter, it is applied to each of the query results. The final result is a list that includes one element per host where each list element is the data object containing the result data object.

Theme Selection

Themes can be defined so that views can have a different appearance when printed. In fact, many variations of print themes can be defined. Note that you must choose the appropriate components. Composite views that use a Fixed layout still print in the sizes that are strictly specified by those views and do not look substantially different than they do when viewed online. For more information, see “[WCFTHEME](#)” on page 156.

There are special dialogs for color or palette selection that are available for end user use.

If the property on which the theme is being set allows a choice of colors, the dialog changes as follows:



Icon Selection

| Property | Description |
|----------|---|
| Color | A chooser allows you to accept colors from a palette. |
| Custom | Allows you to input component, style and value strings. |

Icon Selection is how you specify an Icon runtime value. The Icon Selection allows a user to select a specific icon (as defined in the Icons tab of one of the modules), and attach an Icon Renderer. The Icon Renderer controls the size of icon that is drawn.

An Icon Selection has the following properties:

| Property | Description |
|----------|---|
| Icon | A drop-down list allows you to choose an icon from the existing group of System icons, or ones that have been added to user modules. For more information, see “ Icons ” on page 151. |

| Property | Description |
|----------|--|
| Renderer | A drop-down list allows you to choose a renderer from the existing group of System renderers, or ones that have been added to user modules. See “ Renderers ” on page 148. |

String Template

A String Template Runtime Value evaluates to a string or list of strings. Unlike a simple string, this Runtime Value accepts parameters that can partly determine its value.

A String Template has the following properties:

| Property | Description |
|-----------------|---|
| Value | A text box allows you to type a value for the string. Parameters are permitted. |
| Renderer | A drop-down list allows you to choose a renderer the existing group of System renderers, or ones that have been added to user modules. See “ Renderers ” on page 148. |
| Parameter block | Allows you to choose a Runtime Value for On Null, or any other parameter that has been declared in the Value box. |

Parameters are added to the String Template through standard 0-based arguments common in programming. The first parameter is substituted for the text {0}, the second for {1}, continuing for each sequential number. For example, the *String Name Machine{0}, Application{1}* with parameters that evaluate to *One* and *Two*, returns the value *Name, MachineOne, ApplicationTwo*.

If the parameter evaluates to a list, then the String also evaluates to a list, repeating the string for each value in the list.

For example, suppose the String Name: {0} has a parameter that evaluates to a list of server names:

- MachineOne
- MachineTwo
- MachineThree

Then the String Runtime Value evaluates to the following list:

- Name: MachineOne
- Name: MachineTwo
- Name: MachineThree

A renderer can also be specified for a String Runtime Value.

Rich Text and Rich Text Template

The Rich Text data value is the same String Template, but permits some XHTML in its content. There are no block-level constructors. For more information see [String Template](#). A renderer cannot be chosen. Any formatting must be controlled by the use of XHTML tags.

The Rich Text Template allows parameters to be set as well. A Rich Text Template has the following properties:

| Property | Description |
|-----------------|---|
| Value | A text box allows you to type a value for the string. Parameters are permitted as well as XHTML. |
| Parameter block | Allows you to choose a Runtime Value for On Null, or any other parameter that has been declared in the Value box. |

Image Reference

A reference to an image that is specified by its URL.

Localized String

Note This runtime value is available only for components in system modules.

A Localized String Runtime Value evaluates to a string whose representation may change depending on the locale. This can be a simple string, or it can accept parameters that evaluate to a string. The parameters are similar to a String Runtime Value. When entering a Localized String, there are two input fields:

- Localized String (required)
- Renderer (optional).

The desired Localized String is edited differently depending on whether you are editing a view within a System Module or a User Module. In a System Module, the localized string is typed in, whereas in a User Module it is selected from a drop-down list.

The localizations are stored in a properties file associated with the *wcf.xml* file that defines the module. However, depending on the application, these files may be stored in the file system, or in a database. The English text files are named `strings.properties`.

Other languages have their own set of files. For example, the corresponding French files are named `strings_fr.properties`. If the files required for your language do not currently exist, contact your system administrator about how to copy and edit the existing `strings.properties` files into the desired filename for your language, and install those new files.

The values in the properties files are in the form *key=value*. The key is an internal code used by the system. If you use a key that contains spaces, precede them with a backslash. For example, the key *cpu usage* is entered as *cpu\ usage*. The localized value of the selected string is displayed in the Runtime Value drop-down list tree.

Data

Generally, Query Selection Runtime Values are used to retrieve data from a data source. However, you can use a Data Runtime Value in limited cases to directly select one data object or a list of data objects.

A Data Runtime Value has the following properties:

| Property | Description |
|------------------|---|
| Data Source Type | vFoglight This displays the type as plain text. It is only for your information and cannot be edited. |
| Data Source ID | vFoglight-5 A drop-down menu of the available IDs. You must select a specific data source instance to force the data object to be selected from that instance. |

| Property | Description |
|-------------|---|
| Data Object | A tree of all of the data available for the given Data Source ID is displayed. You select the desired data object by expanding the tree as required, and then clicking on a data element. This could represent a list of data objects, a single data object, or a simple property of a data object, such as a string property. This is somewhat similar to the Path parameter on a Query Selection, or Context Selection, or the Root Path property of a Query. |
| Renderer | A drop-down list allows you to choose a renderer the existing group of System renderers, or ones that have been added to user modules. See “ Renderers ” on page 148. |

A Data Runtime Value cannot have parameters, but it can have On Null.

There are some significant differences between the usage of a Query Selection Runtime Value and a Data Runtime Value. You do not have to specify the Data Source ID for queries to enable them to select from the default Data Source. By contrast, Data Runtime Values do have to have their Data Source ID specified. As a result, they are generally only used for testing or in User Modules, never in System Modules because the Data Source ID in System Modules is not known in advance. Queries are much more flexible and powerful than Data Runtime Values.

Date

A Date object.

List

The List Runtime Value must always have parameters. It allows you to either make a list containing the values of each of the parameters, or to make a list that merges all elements in the values of the parameters into one list.

A List Runtime Value has the following properties:

| Property | Description |
|-------------------|---|
| Merge Lists | <p>If unchecked, the resulting list contains one element for each parameter, with the element containing that parameter's value.</p> <p>If checked, all elements in the list values of the parameters are merged into one big list. In addition, if the parameter values are lists of lists, only the bottommost non-list elements are put into the final list, so that no element of the final list is itself a list. Any data object that is seen as a duplicate of an existing data object is eliminated. However, if the objects in the list are not data objects, duplicates are not eliminated.</p> |
| Remove Duplicates | <p>Only applies when Merge Lists is true. If true, then duplicates are removed from the merged list, otherwise they are not. A special option, <i>Intelligent</i> is available for List Runtime Values created in older versions of the <i>Web Component Framework</i>, in which nulls were unpredictably removed or not.</p> |
| Remove Nulls | <p>If true, nulls are removed, otherwise they are not. A special option, <i>Intelligent</i> is available for List Runtime Values created in older versions of the <i>Web Component Framework</i>, in which nulls were unpredictably removed or not.</p> |
| Renderer | <p>A drop-down list allows you to choose a renderer the existing group of System renderers, or ones that have been added to user modules. See “Renderers” on page 148.</p> |
| Parameter block | <p>Allows you to choose a Runtime Value for On Null, or any other parameter that you add using <i>Add Parameter</i>.</p> |

Return Types

There are ten possible types of data that you can extract from the context and the context paths:

- [Localized Value](#)
- [Value](#)
- [CountPerItem](#)

- [Count All](#)
- [Unit](#)
- [Units](#)
- [Localized Property Name](#)
- [Localized Property Names](#)
- [Localized Type Name](#)
- [Localized Type Names](#)

Localized Value

This is the default return type. It returns the basic type of data of the context, but localizes it if a localization is available. If the context is a host server and the path is name, then it returns a simple string. If the context is a list of servers and the path is name, then it returns a list of names.

Value

This is the default return type. It returns the basic type of data of the context, not its localized value. If the context is a host server and the path is name, then it returns a simple string. If the context is a list of servers and the path is name, it returns a list of names.

CountPerItem

This is used when you want to show the count of a list. For example, use this property to display the number of file systems on a host. Count acts on the result of evaluating the Context Selection Runtime Value normally. There are several cases:

- If the result is not a list, then the number of objects is “1,” and therefore the count is “1.” This is not a common use of this property. If you set up the context to be a single value, then you do not need to derive the count.
- If the result is a list of objects, then this returns the number of items in the list. For example, a Host data object has the events property, which is a list. If this property is used as the context value, then Count returns the number of events.
- If the result is a list of lists, then it returns a list composed of the results of evaluating the *CountPerItem* on each value in the top level. For example, if the context is set to hosts, and the path is set to events, then *CountPerItem* returns a list of the number of events for each host.

- If the result is a list of lists of lists (or even deeper), the rule is the same as in the previous case, but it can return lists of lists (and so on...) of counts.

Count All

Unlike *CountPerItem*, this return type just counts up all of the non-list elements of a list. For example, if context is set to Hosts and the path is set to Events, then Count All returns the total number of events for all of the Host data objects. *CountPerItem* would return a list, with each item a count of events under a given host.

Unit

This returns the unit associated with the path. If the path is set to *cpuUsage*, then this returns the unit object for percentage (%). If the path is set to *diskUsage*, then this returns the unit object for size (in MB). Unit returns null if no unit is associated with the path.

Units

Unit only returns one unit, even if the path is set to a list property. If you have a view that contains a list of different metrics for one object, then you can use Units. This returns a list containing the unit for each item. For example, the view shows the CPU usage and the disk usage for a host in a row-oriented table. Set the Return Type to Units for a column, and the entries shows % and MB for CPU usage and disk usage, respectively.

Localized Property Name

Each property of a data object has a localized name. This Return Type returns the localized name of the property set in the Path. For example, if the context is a Host, and the path is set to *cpuUsage*, then Localized Property Name returns CPU Usage.

Localized Property Names

Similar to Units, this returns a list containing the localized name for each element of a list. This differs from Localized Property Name, which returns one value for the first element in a list.

Localized Type Name

This returns the localized name of the type of the property. For example, if the item taken from the context is a Host, and the path is set to *cpuUsage*, then Localized Property Name returns Metric. If the path is set to name, then Localized Type Name returns String.

Localized Type Names

Similar to Localized Property Names, this returns a list containing the localized name of the type of the property for each element of a list.

Additional Components

A number of helper components round out the Web Component Framework. Renderers are available to provide special formatting. The Task mechanism allows you to launch actions from your view. You can File arbitrary Icons and use them in your views. Units are available for use with metrics. Theme and Module Resources let you specify different appearances, such as the Application theme or the Monitoring theme. The Printing mechanism allows you to print reports.

This chapter contains the following sections:

| | |
|--|-----|
| Renderers | 148 |
| Tasks | 150 |
| Icons | 151 |
| Files | 152 |
| Types | 153 |
| Units | 154 |
| Theme and Module Resources | 155 |
| Printing | 156 |

Renderers

Setting Renderers, Icons and Units in Views

Configuring how data is displayed in a dashboard is determined in the view definition, but relies on the existence of renderers, icons and mappings that are configured separately from the view.

Renderers Tab

Renderers are created and configured in the Renderers tab in the Module Contents pane. Renderers determine the display of data. For example, the number of decimal places shown by a value inside a table cell is decided by a Number Renderer instance, the display of data as an icon is determined by an Icon Renderer instance, and the display of a metric as a time plot chart is configured using a Time Plot Renderer instance.

At runtime, data in a vFoglight dashboard displays depending on which renderers are set in the view definition and which renderer has been mapped to a data value's type, property or unit. Renderers can be mapped to types and properties in the Types tab or to units in the Units tab. Mapped renderers are looked up automatically at runtime if no specific renderer has been set in the view definition.

Overriding a Default Renderer

All the units supported by vFoglight Core are associated with a default renderer. These associations can be found in the Core: System module under the Units tab. For example, a percent unit is associated with the Percent Renderer.

To define a new renderer:

- 1 In the Module List pane, select the module that should contain the renderer.
- 2 Select the **Renderers** tab in the Module Contents pane.
- 3 Click the **Add** button and choose either a **Blank renderer** or a **Copy of** an existing renderer.

If you choose a blank renderer, select one from the **Type** drop-down list.

A **New Renderer** tab appears in the Definitions pane.

- 4 In the **General** tab, give the new renderer a name and supply an appropriate description.

- 5 Select the **Configuration** tab and set the appropriate properties for the new renderer.
- 6 Click **Save**.

The new renderer is added to the selected module.

Note You can define a new renderer to override the default settings an existing renderer. For example, the default text renderer has the Support Newlines property set to false. If you need newlines, set the property to True. The cost is a slight slowdown in rendering.

To associate the renderer with a data type:

- 1 In the Module List pane, select the module that should contain the renderer.
- 2 Select the **Types** tab in the Module Contents pane.
- 3 Click the **Add** button.

The *New Type Mapping* dialog appears.

- 4 Select the desired data type in the drop-down list, and then click **OK**.

A **New Type Mapping** tab appears in the Definitions pane.

- 5 Select its **Renderers** tab.
- 6 In the left pane underneath this tab, select the same renderer type that you chose in the previous procedure.
- 7 In the **Renderer** drop-down list, select from the Available Renderers dialog the renderer that you defined previously. Click **Save** in the dialog.
- 8 Click **Apply** in the editor pane.
- 9 Click **Save**.

Note If you have a requirement to use different renderers for different sorts of labels, say in table columns, you can use the table's nested grouping feature. Since columns are independent of groups, you can assign a renderer to one column and a different renderer in another column. This avoids the complexity of having to use one renderer and parameterizing it, which may cause alignment problems in the table's columns.

Tasks

Tasks are logical actions external to *Web Component Framework*. Examples in the vFoglight regime are tasks for clearing and acknowledging alarms, building a service, scheduling a report, or launching PerformaSure in context on a particular request.

Otherwise tasks are configured to behave in the flow in the same manner as views.

The workflow generally looks like:

- Action on a View
- Invoke component
- Update page

Some tasks may break the flow – that is, not return. An example of this is launching PerformaSure. In this case the task does not have a flow action that can be configured to update the page.

A task consists of its code and three text files, and is usually packaged in a JAR file.

Components of a Web Component Framework Task

| | |
|---------------------------|---|
| <i>descriptor.xml</i> | Location of the code for the task and its characteristics. |
| <i>types.xml</i> | Configuration schema: the task's properties. |
| <i>strings.properties</i> | The English text for the component. This provides a default localization. Other localization files have names like <i>strings_fr.properties</i> . |

Tasks are located in *META-INF/wcf-metadata/task/X.Y*, where *X.Y* is the Web Component Framework internally generated name for the component.

Note You can use the Execute Groovy Task component to implement a simple task. See the *Web Component Tutorial* for an example.

Icons

The Icons tab allows you to configure how graphics render in vFoglight. In a view definition, an icon can be referenced directly by using an Icon Selection Runtime Value and then setting an Icon Renderer to specify the desired size. In the Types tab, an icon can be mapped to a data type. In a view definition, selecting data of that type and specifying an Icon Renderer causes the data to display as the mapped icon.

Icons collect differently-sized (but similar in appearance) images under a single name. Images can be specified for the following sizes:

- Extra small (8x8 pixels)
- Small (11x11 pixels)
- Medium (16x16 pixels)
- Large (32x32 pixels)
- Extra large (64x64 pixels)
- Huge (128x128 pixels)
- Scalable (any size)

Types Tab

Mappings of icons to data types and mappings of renderers to data types and properties are configured on this tab. When an icon is mapped to a data type, the icon is displayed if an Icon Renderer is specified on any View Property referencing data of that type. When a renderer is mapped to a data type or property, any view property bound to data of the type or property is displayed using the mapped renderer unless a specific renderer is set on the view property.

The *Web Component Framework* has a concept of renderers for Runtime Values, which can be used to format the output from the *RuntimeValue* in various ways. Definitions for all of the renderers are held in a one-file-per-*wcf.xml* file and in a global file that is common to the whole application. The files are named *renderers.xml*.

The ID attribute is a unique name that is associated with the corresponding renderer. This is the value that is to be specified in the renderer-ID of *RuntimeValue*. See “[Data](#)” on page 141.

The optional `<associations>` section specifies associations between the renderer and *sdo* types, metric, classes or units, depending on the tags it contains. Each of the tags in the associations section is optional and more than one association of each type may be

specified. These associations are used to resolve the renderer when a Runtime Value is rendered.

The `<configuration>` section can be any of the types of configuration mentioned previously. However, each specific class of renderer expects a specific type of configuration.

Files

Users can upload files such as image files to each module's public directory, by using the Files tab in the Definition Editor. Files can be uploaded to the root directory (under public), the images directory, or other subdirectories.

To manage files within a module:

- 1 Select the module then click the **Files** tab.

The list of files (with their paths) within that module is displayed. Clicking on a filename displays the filename along with the special *Web Component Framework* URL that refers to that file, for instance in the Background Image field of a View, the URL field of an Image Renderer, or the Icon field of a State Renderer.

You can copy that URL to the clipboard for later use.

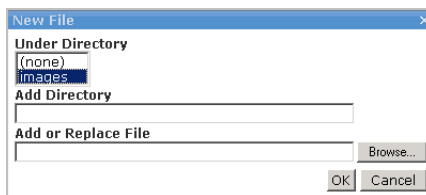
- 2 Select the URL, then click **Edit > Copy** into the command line of your web browser.

To delete a file:

- 1 Select the file, then click the **Delete** toolbar icon.
- 2 Select **OK** to confirm the deletion of the file.

To add or update a file:

- 1 Click the **Add** toolbar icon. The following dialog box appears:



You can select which existing directory (under public) the file is uploaded to by selecting from the Under Directory list box. You also move it into another subdirectory of the selected directory.

- 2 Type the new subdirectory name in the **Add Directory** field.
- 3 Type in the full path name on your machine for the file to be uploaded in the **Add or Replace File** field.

Alternatively, click the **Browse** button to select the file using a standard file open dialog box.

- 4 Click **OK**.

A confirmation message appears confirming the upload that is going to happen. When you click OK on that message, the upload occurs. Unless an error message is shown, the new file displays in the list of files, along with the *Web Component Framework* URL for accessing it.

Types

Type mappings associate entities such as renderers, icons, and flows to a specific data type or data type property. At runtime, *Web Component Framework* will look up the entities mapped to these types and properties to use as defaults if more specific entities have not been set.

If you configure a view's flow with a flow type of **Select type flow**, the *Web Component Framework* looks in its type mappings at runtime for a flow associated with the type in question.

You can register different kinds of default flows by specifying tags. You can provide your own by typing its name into the tag name combo box, or choose one of the predefined tags:

- *drilldown*—used for a flow that takes you to a new page.
- *summary*— used to retrieve a view appropriate for a dwell.
- *menu*—used to retrieve a view that gives the user a list of options with a temporary popup.
- *dialog*—used to flow to a view that is sized to appear overtop of current data, either as a temporary or permanent popup as appropriate.
- *edit*—used to flow to the appropriate editor for the specified object.

At runtime, if no specific renderer has been specified on a runtime value, the *Web Component Framework* will look in the type mappings to see if a renderer has been mapped to the type of the data or the property the data was retrieved from. In the event of a conflict, renderers mapped to properties take precedence over renderers mapped to types.

When an Icon Renderer is explicitly specified as the renderer on a runtime value, the renderer looks in the type mappings to find the icon that has been mapped to the type of the data evaluated from the runtime value.

Units

Unit mappings associate renderers to a specific data unit. At runtime, if no specific renderer has been specified on a runtime value and no renderer has been found in the type mappings, the *Web Component Framework* looks in the unit mappings to see if a renderer has been mapped to the unit of the data.

Mappings of renderers to units are configured in the Units tab. When a renderer is mapped to a unit any view property bound to data of the unit displays using the mapped renderer. This is unless a specific renderer is set on the property or another renderer is mapped to the data's type or property.

Determining the Appropriate Renderer for a Runtime Value

Determining which renderer to use for a given *RuntimeValue* involves several possibilities. The following are the current possibilities, listed in priority order, which means if the first one does not apply, the second one is tried, and so on.

- If a named renderer is given for this *RuntimeValue* (in the `renderer-id` attribute), use it.
- If the value is a *Web Component Framework* `DataObject`, and a renderer is associated with its `SDO Type`'s name (via a `<type>` element within the `<associations>` element), use that renderer.
- If the value is derived from a metric, and is a `Number`, and a renderer is associated with the fully qualified name property of the metric (via a `<metric>` element within the `<associations>` element), use that renderer. Metric names are fully qualified as `datasource name`, a colon and the metric name, for example *cpuUsage*

- If a renderer is associated with the fully qualified class name of this object (via a `<class>` element within the `<associations>` element), for example, *java.lang.Double*, use that renderer.
- If a renderer is associated with the fully qualified name property of the unit of this object (via a `<unit>` element within the `<associations>` element), use that renderer. The unit name is fully qualified by preceding it with the `DataSource` name and a colon, for example `foglight-4.2:percent`.
- If none of the preceding rules apply, use the Default Renderer.

Several methods have been added to the `RuntimeValue` class related to renderers. The first pair return `DataPoints` instead of `Objects`. A `DataPoint` is composed of the evaluated value of the `RuntimeValue`, plus its renderer.

Theme and Module Resources

The ability to refer to files through the theme or in the module is most useful in specifying images for state renderers or creating icons.

Themes can contain files (most usually images) which require separate versions for different themes, locales or UI sizes – an example would be an image that has a 16x16 version for `SMALL` sub-themes, a 32x32 version for `NORMAL`, a 64x64 version for `LARGE`, and a dark-background version for the normal UI but a black-on-white version for the printable theme.

Modules can also contain files (in a public repository) which are available to the application's users. It is also possible to create a theme directory structure within a given module and store theme-specific files for selective access.

There are three types of resource path:

- [WCFTHEME](#):
- [WCFMODULE](#):
- [WCFMODULETHEME](#):

Note In the following, always use the default images directory as the root for any image files that you add. Custom paths are not supported.

WCFTHEME

This theme is converted into a theme request on the basis of the current theme settings, and the request for the specified file is handled by the ThemeManager. For example, *WCFTHEME:images/test.png* with a current theme of default, locale of en_US and size of NORMAL would become a request to *{context}/themes/default/en_US/NORMAL/images/test.png* and would be handled by the ThemeServlet.

WCFMODULE

The resources path requires the specification of a module name to determine where the look-up is to be performed. Paths beginning with this prefix are handled by the standard module access (persistence) mechanism, accessing the named module's public files. In this case, the path *WCFMODULE:global/images/test.png* would be mapped to the path *public/images/test.png* in the global module and passed to the persistence service to be resolved.

WCFMODULETHEME

This theme combines the previous cases. The prefix requires the specification of a module name as an argument, and then assumes the existence of a theme structure in the module's public files against which a theme lookup is performed using the current theme. Assuming our theme settings are *monitoring_default, fr_CA*, and we are given the path *WCFMODULETHEME:Core:system/images/bg/sample_bg.png*, then the path will be rewritten as *public/themes/monitoring_default/fr_CA/LARGE/images/bg/sample_bg.png*.

Printing

There are three different mechanisms available for printing in the *Web Component Framework*:

- [Web Browser Printing](#)
- [PDF Generation](#)
- [Reports](#)

Web Browser Printing

It is possible to print from a web browser either by using your browser's print command, or by using the print option in the action panel. Using the browser print function prints the page as it appears.

Note Printing a dashboard in Firefox is not supported.

When the page prints, only the informational details of the page are displayed. Navigation controls are hidden from the printable view in order to show as much detail as possible.

- The page uses the theme that you have selected in your user preference.
- Components are re-rendered in a print-friendly fashion. For example, tables that limited their height now render all their rows to the printed page.

Note Container views that use a Fixed layout print in the sizes that are strictly specified by those views and do not look substantially different than they do on the page.

PDF Generation

To produce a printable version of a view in PDF format you need to create a report. Simple reports can be produced by selecting **Create Report** in the action panel. Drag in the views you want and click **Create PDF**. See the *vFoglight User Guide* for additional details. Custom reports can be produced using the Report component. See the *Web Component Tutorial* for instructions on using this component.

One of the major advantages of using PDF printing over browser printing is that PDF printing renders a chart as lines so that they can be printed with the full resolution of your printer, where browser printing limits the resolution of charts to that of the screen.

Another advantage of PDF printing is that views built with the Report component have the ability to have page-friendly features such as custom headers and footers, which include features such as page numbers and titles. In addition, it is possible to control the orientation of each page, and where pages break.

Reports

Reports are useful both for informative and archival purposes. You can use the Report container to structure a group of views that you deem useful for dissemination to interested parties or simply to keep as a historical record.

Reports in PDF format may be generated in color or in black and white (monochrome).

The Page Decoration component is used to define headers and footers in a report layout if the intention is to convert the report to PDF format. If you use them to define a header and a footer for a report, both header and footer elements must be placed before the body pages. This design is required so that footers will appear in the case where the body component is a table whose rows may span several pages.

It is possible to arrange for different headers and footers after the first page, and in different sections of a large report. Simply add more Page Decoration components after body views whenever new ones are required. You can emit a page break both before and after any view, which gives added control over the layout, such as permitting you to start a new section on a fresh page.

See the *Web Component Reference* pages on Report and Page Decoration for detailed descriptions of these components, or refer to the *Web Component Tutorial*.

Report Layout

The Report component is designed to permit multi-page reports. It does this by checking if there is enough space for the next view, and if there is not, generating a new page. Reports also have special view types of header and footer, which are used to generate running headers and footers on the pages of the reports. A footer must be specified before the view on the first page it is to appear because it must be available when the page is generated in order to allocate the correct amount of space for it. Headers and footers can be changed later on in a report by re-specifying them among the body views of the report.

Once the report determines where on a page to place a view, that view is drawn in the report's graphics, as opposed to the report generating a graphics for the view to use. This means that reports which are nested in a parent report can also span multiple pages, but it also means that the report is effectively transparent, and can not have a background or border. Nested reports can not specify headers or footers—only the top-level report is able to create them.

Note Container views that use a Fixed layout print in the sizes that are strictly specified by those views and do not look substantially different than they do on the page. Most tables, either printed directly as a report or embedded in a report, are able to flow to more than one page.

For this to work, the table can only be nested in some combination of reports and iterators. In all other cases (for instance, a table in a fixed or grid layout) the table has a maximum size of a single page, and will clip to that.

Scheduling Reports

It is possible to set up a schedule in the **Administration** module under *Dashboards* in the navigation panel so that reports are generated automatically at regular intervals. See the vFoglight Administration and Configuration Guide for details on setting up a schedule and associating a report generation task with it.

Remote Access to Views

This section describes the mechanisms that are available for embedding a *Web Component Framework View* in your application. The following are supported:

- [Portlet](#) (JSR 168)
- [Google Gadget](#)
- [SharePoint Web Part](#)

Portlet

This is a JSR168 compliant portlet created for the purpose of allowing remote access to browser interface's views. So far, this portlet has only been successfully tested on these Portal Servers as shown below:

- JBoss Portal Server 2.6
- Apache Pluto 1.0.1
- WebLogic Portal 10

Deploy and Run

In order to test your vFoglight portlet, you will need to have a portal server available in which you can place the portlet's *war* file. JBoss Portal, Apache Jetspeed2, and WebLogic Portal are suitable choices for testing a remote portlet.

To test your portlet:

- 1 Locate the vFoglight remote portlet *war* file:
`${vFoglight Server Home}/tools/foglight-remote-portal.war`

- 2 Deploy the portlet.

The specific steps depend on which portal server you use. Please follow the guidelines given in your particular portal server for deploying portlets. For instance, to deploy to Pluto and JBossPortal Server, the simplest way is to simply copy the *war* file into the server's deploy directory.

- 3 Access the portlet.

Once the portlet is successfully deployed, be sure to follow the user guide of your portal server for information on how to access its deployed portlets.

Configuration

You need to configure the portlet by setting its host, port, and viewId before attempting to run it.

- host—the vFoglight Server host you would like to connect with.
- port—the vFoglight Server port you would like to connect with.
- viewId—the *Web Component Framework* fully qualified Id, for example, *system:fsmhome.0*. There are also predefined view mappings available for use.

Note Validations for all these settings are also available.

Google Gadget

A vFoglight view can be added to your Google portal. For information on creating your own Google pages, follow the link given [here](#). Once you have your Google page you can add a vFoglight view to it.

To edit your Google page to contain a vFoglight view:

- 1 Two vFoglight files are required, *foglight-remote.xml* and *ALL_ALL.xml*.
These files are located in `<vFoglight Server Home>/tools/gadgets/foglight-remote-google-gadget`.
- 2 Sign in to your Google site.
- 3 Upload the two files listed in step 1 to your Google site. Google reports that their site is a work in progress, so the instructions given here are of necessity general in nature. At the time these procedures were written, the upload link was located on the Site Manager page for your Google page.
- 4 Switch to your Google page and click the link to add a Gadget.
A dialog called *Add a gadget to your page* should appear.
- 5 Click **Add by URL**. Enter **`http://<yourPage>.googlepages.com/foglight-remote.xml`** and then click **Add**.
A setup dialog appears.
- 6 Type the name of the vFoglight server in the Server/Host text box.
- 7 Type a view name in the View Id box. You can type the reference ID of a view, which is found by selecting a view in a regular browser interface and clicking

Properties in the general tab of the right action panel, or you can type the name of one of three preconfigured views:

- Alarms
- Hosts
- Services

8 Test your page.

SharePoint Web Part

vFoglight Remote Web Part, which has been developed to allow remote access to a vFoglight browser interface, is compatible with Microsoft SharePoint Web Part. There are two versions, one for ASP .NET 1.1 and the other for ASP .Net 2.0 runtime environments.

To edit your Microsoft SharePoint Web Part page to contain a vFoglight view:

1 Locate the appropriate *CAB* file.

For ASP .NET 1.1, use `<vFoglight Server Home>/tools/foglight-sharepoint-webpart-dotnet-1.cab`

For ASP .NET 2.0, use `<vFoglight Server Home>/tools/foglight-sharepoint-webpart-dotnet-2.cab`

2 Add the Web Part package to your virtual server.

For instructions, see <http://technet.microsoft.com/en-us/library/cc288254.aspx>.

3 Register the Web Part to the Web Part gallery of the site collection.

For instructions, see <http://technet.microsoft.com/en-us/library/cc287891.aspx>.

4 Configure the Web Part by setting the vFoglight fields for **host**, **port** and **ViewId**.

- **Host**—The IP address of the vFoglight server.
- **Port**—The port number for the vFoglight server.
- **ViewId**—The fully qualified Reference Id for the view component you are going to add, which is found by selecting a view in a regular browser interface and clicking Properties in the general tab of the right action panel, or you can type the name of one of three preconfigured views (system:fsmhome.0, Alarms, Hosts, Services).

Index

A

- about vFoglight 8
- aggregations 82
- allowed role 78

C

- charts 47
- choose type flow type 120
- choose value flow type 119
- comments
 - in a view 112
- comments field
 - in a query 78
- common layouts 44
- comparison
 - in a query 88
- conditional types
 - in a query 87
- configuration tab
 - in a view 113
- contacting
 - Vizioncore 12
- containers
 - available layouts 45
- context 62
- context help
 - in a view 112
- context help field
 - in a query 78

context inputs

- in a view 113

context selection 133

context tab

- in a view 112

cooperative layout 114

copy

- entity 57

- view 103

custom purposes

- in a view 112

D

dashboard

- context selection 133

- managing files 152

- printing 156

- themes and modules 155

dashboard page 70

data source

- id 80

- type 79

data sources tab 71

deep copy

- view 106

default values 64

definitions 57

definitions and entities 57

definitions menu 105

definitions page 72

deleting

query 99

documentation 9

cartridge 10

core 10

feedback 11

suite 9

E**editing**

query 98

editor

for web components 66

entities 57**entity**

copying 57

public 57

enum 62**evaluation sequence**

in query conditions 91

external URL flow type 120**F****files** 152**filter results**

in a query 85

flow

configuring 115

configuring in a view 115

type 115

flow tab

in a view 114

flow type

choose type 120

choose value 119

external URL 120

invoke task 120

next page 116

popup 116

previous 117

select tagged view 117

select type flow 117

sequence 117

show help 121

update 115

G**gauges** 47**general tab**

in a view 108

H**hide root checkbox** 77, 82**I****identifying values**

in an aggregation 84

invoke task flow type 120**is set**

in a query 89

L**layouts** 45**M****metric** 59**module** 155**module-level roles** 79**modules**

validating 56

N**name**

in a query 77

nested views 106**next page flow type** 116**not**

in a query 90

null values 63

O

object type 81

observations

- and time range 58
- enum 61
- metric 59

P

parameters

- in queries 94
- in runtime values 63

paths 65

popup flow type 116

previous flow type 117

procedure

- add a case 120
- add a not condition 90
- add a required parameter 80
- add an aggregation 83
- add an and condition 90
- add an Identifying Value 85
- add an or condition 90
- add an Order By key 86
- add or update a file 152
- associate the renderer with a data type 149
- copy a query 96
- create a container view 102
- create a new query 91
- create a new view based on a copy of an existing view 104
- create a query 95
- define a new renderer 148
- define the flow for a view 118
- delete a file 152
- delete a query 99
- derive a query 98
- edit a query 98

manage files within a module 152

open a definition 72

query selection 136

remove a case 120

remove a entire where clause 87

remove an Identifying Value 85

remove an Order By key 87

set the choose action 119

test your portlet 160

public 57

purpose

- in a view 110
- of a view 110

Q

queries

- configuring 75
- creating 91
- definition settings 77

query

- copying 96
- creating 94
- deleting 99
- editing 98
- overview 76
- root 77

R

relevant role 78

remove

- view 105

removing

- multiple databases 71

renderers 63

replace

- view 107

required parameters

- in a query 80

roles

- for a module 79

- in a view 112
- in queries 78
- in views 78

root path 81

root query 77

runtime values

- for pages and views 130

S

save

- view 105

select tagged view flow type 117

select type flow type 117

sequence 117

sequence flow type 117

sequence of evaluation

- in query conditions 91

show help flow type 121

sub type is

- in a query 90

suite 9

system dashboard 70

T

tables 47

tabs

- in the view definition pane 107

text conventions 11

theme 155

Threshold 53

time range 48

topology 49

trees 47

types

- conditional, in a query 87
- in flows 115

U

update flow type 115

user dashboard 70

V

validating modules 56

view

- charts and gauges 47

- commands 105

- common 44

 - Image* 44

 - Key-Value Listing* 44

 - Label* 44

 - Row-Oriented Table* 44

 - Time Plot Chart* 44

 - Time Range Zonar* 44

- configuration tab 113

- configuring 102

- copying 103

- creating 102

- creating a new container 102

- custom purpose 112

- deep copying 106

- flow tab 114

- nested 106

- new view based on a copy 103

- purpose 110

- removing 105

- replacing 107

- roles 112

- saving 105

views

- charts and gauges 47

 - Bar Gauge* 47

 - Bar or Pie Chart* 47

 - Chart Legend* 47

 - Circular Gauge* 47

 - Cluster Bar Chart* 47

 - Cylinder* 47

 - Pulse Gauge* 48

 - Time Bar Chart* 48

 - Time Plot Chart* 48

- Time Range Zonar* 48
- TimeState Chart* 48
- configuring 73
- containers
 - Column Layout* 45
 - Fixed Layout* 45
 - Form* 45
 - Grid* 45
 - Iterator* 45
 - Report* 45
 - Splitter* 45
 - Stack Layout* 45
 - Switch* 46
 - Tab Container* 46
 - Topology* 46
 - Type* 46
 - Wizard Layout* 46
- inputs 49
 - Button* 49
 - Check Box* 49
 - Context Inputs Editor* 49
 - Date/Time Input* 50
 - Drop-Down List* 50
 - Filter* 50
 - Number Input* 50
 - Radio Button List* 50
 - Text Area* 50
 - Text Field* 50
 - Time Range Drop-Down* 50
 - Time Range Form* 50
- others 51
 - Filter* 51
 - iFrame* 51
 - Include* 51
 - Links Box* 51
 - List Viewer* 51
 - Page Title* 51
 - Progress* 51
 - Property Viewer* 51
 - Separator* 51
 - Syndication Feed* 51
 - Toolbar* 51
 - Wizard Layout* 51
- purposes 110
- renderers 52
 - Date* 52
 - Error* 52
 - Host Name* 52
 - Icon* 52
 - List* 52
 - Null Image* 52
 - Null String* 52
 - Number* 52
 - Number Bar* 52
 - Number Percent* 52
 - Number Unit* 53
 - Pulse Gauge Renderer* 53
 - Range* 53
 - Sparkline* 53
 - State* 53
 - String* 53
 - Threshold* 53
 - Time Range* 53
- reporting 49
 - Page Decoration* 49
 - Report* 49
- tables and trees 47
 - Array Table* 47
 - Cell-Oriented Table* 47
 - Key-Value Listing* 47
 - Row-Oriented Table* 47
 - Tree Table* 47
- time range 48
 - Minuscule* 49
 - Time Range Drop-Down* 48
 - Time Range Form* 48
 - Time Range Zonar* 48
 - Title Only* 49
 - Topology* 49
- topology 49

W**Web Component Framework**

about 29

where clause

in a query 87