



Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide

Corporate Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA

<http://www.cisco.com>

Tel: 408 526-4000
800 553-NETS (6387)

Fax: 408 526-4100

Text Part Number: OL-1279-01

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

The following information is for FCC compliance of Class B devices: The equipment described in this manual generates and may radiate radio-frequency energy. If it is not installed in accordance with Cisco's installation instructions, it may cause interference with radio and television reception. This equipment has been tested and found to comply with the limits for a Class B digital device in accordance with the specifications in part 15 of the FCC rules. These specifications are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

Modifying the equipment without Cisco's written authorization may result in the equipment no longer complying with FCC requirements for Class A or Class B digital devices. In that event, your right to use the equipment may be limited by FCC regulations, and you may be required to correct any interference to radio or television communications at your own expense.

You can determine whether your equipment is causing interference by turning it off. If the interference stops, it was probably caused by the Cisco equipment or one of its peripheral devices. If the equipment causes interference to radio or television reception, try to correct the interference by using one or more of the following measures:

- Turn the television or radio antenna until the interference stops.
- Move the equipment to one side or the other of the television or radio.
- Move the equipment farther away from the television or radio.
- Plug the equipment into an outlet that is on a different circuit from the television or radio. (That is, make certain the equipment and the television or radio are on circuits controlled by different circuit breakers or fuses.)

Modifications to this product not authorized by Cisco Systems, Inc. could void the FCC approval and negate your authority to operate the product.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

Mobile Wireless Fault Mediator (MWFM) architecture is based on RiverSoft NMOS(tm) and RiverSoft Fault Manager technologies adapted to Cisco's Mobile Wireless environment.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

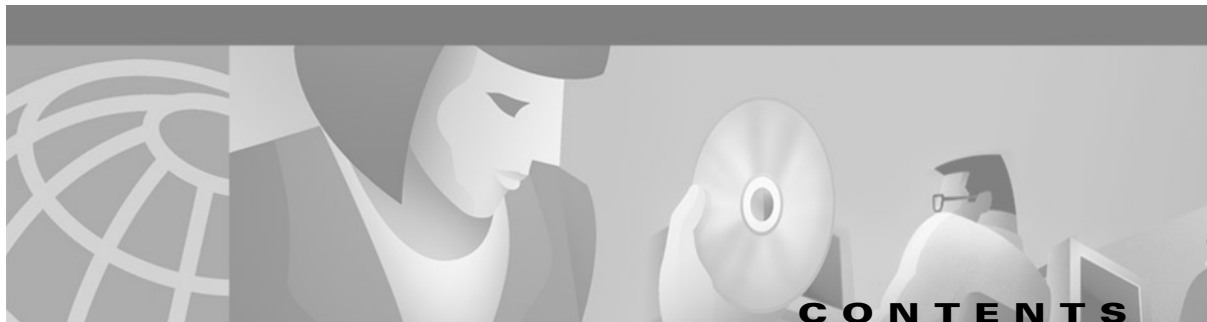
IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

AccessPath, AtmDirector, Browse with Me, CCIP, CCSI, CD-PAC, *CiscoLink*, the Cisco *Powered* Network logo, Cisco Systems Networking Academy, the Cisco Systems Networking Academy logo, Fast Step, Follow Me Browsing, FormShare, FrameShare, GigaStack, IGX, Internet Quotient, IP/VC, iQ Breakthrough, iQ Expertise, iQ FastTrack, the iQ Logo, iQ Net Readiness Scorecard, MGX, the Networkers logo, *Packet*, RateMUX, ScriptBuilder, ScriptShare, SlideCast, SMARTnet, TransPath, Unity, Voice LAN, Wavelength Router, and WebViewer are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn, Discover All That's Possible, and Empowering the Internet Generation, are service marks of Cisco Systems, Inc.; and Aironet, ASIST, BPX, Catalyst, CCDA, CCDP, CCIE, CCNA, CCNP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, the Cisco IOS logo, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Enterprise/Solver, EtherChannel, EtherSwitch, FastHub, FastSwitch, IOS, IP/TV, LightStream, MICA, Network Registrar, PIX, Post-Routing, Pre-Routing, Registrar, StrataView Plus, Stratm, SwitchProbe, TeleRouter, and VCO are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and certain other countries.

All other trademarks mentioned in this document or Web site are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0106R)

Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide
Copyright © 2001, Cisco Systems, Inc.
All rights reserved.

RiverSoft



CONTENTS

CHAPTER 1

Introduction 1-1

- What is contained in this guide? 1-1
- Who is this guide for? 1-2
- Where should I start to read this book? 1-2
- Why use an Application Programming Interface? 1-2
- Styles used in this guide 1-3
 - Note on command line examples 1-3
 - What typographic conventions and symbols mean 1-3
- Summary 1-3

CHAPTER 2

MWFM TIBCO/Rendezvous 2-1

- How TIB/Rendezvous is utilized within MWFM 2-1
- The Rendezvous process 2-1
- How to create a new rvd session 2-2
 - Initializing a rvd session via the NMOS Java API 2-2
- How to create a new rva session 2-2
- Initializing a rva session via the NMOS Java API 2-3
- Summary 2-3

CHAPTER 3

Examples 3-1

- The NMOS Java API test application 3-1
- Starting a Rendezvous rvd session 3-1
 - Obtaining all class records stored in riv_class 3-4
- Querying riv_model 3-6
- Inserting an entry into riv_model 3-10
- Modifying an entry in riv_model 3-11
- Deleting an entry from riv_model 3-12
- Adding and removing a listener for updates from riv_class 3-13

The NMOS Java API CRivClientHelper test applications **3-15**

- Initializing the application, the Rendezvous session and the CRivClientHelper object **3-15**
- Obtaining all event records from riv_f_amos **3-16**
- Obtaining event records from riv_f_amos that match a list of criteria **3-17**
- Clearing an event from riv_f_amos **3-18**
- Obtaining all users from riv_auth **3-18**
- Inserting a new user into riv_auth **3-20**
- Inserting a new user profile into riv_auth **3-21**
- Modifying an existing profile held in riv_auth **3-23**
- Determining if a user has permission to clear an event **3-24**
- Closing the session **3-24**

Summary **3-24**

CHAPTER 4

The NMOS Java API Classes 4-1

Chapter organization **4-1**

- Class Java—The name of the class. **4-1**
- Hierarchy **4-1**
- Description **4-1**
- See Also **4-1**
- Field/Constructor/Method Summary **4-2**
- Field/Constructor/Method Detail **4-2**
- Index **4-2**
- Summary of classes **4-3**
- Class/Interface reference pages **4-6**

Class ASCII_CharStream **4-8**

- Hierarchy **4-8**
- Description **4-8**
- See Also **4-8**
- Field summary **4-8**
- Constructor summary **4-9**
- Method summary **4-9**

Class CRivASN1Address	4-10
Hierarchy	4-10
Description	4-10
Field summary	4-11
Constructor summary	4-11
Method summary	4-11
Field detail	4-12
Constructor detail	4-12
Method detail	4-14
Class CRivAtom	4-16
Hierarchy	4-16
Description	4-17
See Also	4-17
Constructor summary	4-17
Method summary	4-18
Constructor detail	4-19
Method detail	4-21
Class CRivClient	4-26
Hierarchy	4-26
Description	4-27
See Also	4-27
Field summary	4-27
Constructor summary	4-27
Method summary	4-28
Field detail	4-29
Constructor detail	4-29
Method detail	4-29
Class CRivClientHelper	4-37
Hierarchy	4-37
Description	4-37
See Also	4-37
Constructor summary	4-38
Method summary	4-38
Constructor detail	4-39
Method detail	4-40

Class CRivClientInterface	4-55
Hierarchy	4-55
Description	4-55
Inner class summary	4-55
Constructor summary	4-55
Method summary	4-56
Inner class detail	4-56
Constructor detail	4-57
Method detail	4-58
Inner Class CRivClientInterface.RCIRecordListener	4-61
Hierarchy	4-61
Description	4-61
Enclosing Class	4-61
Constructor summary	4-61
Method summary	4-62
Constructor detail	4-62
Method detail	4-62
Inner Class CRivClientInterface.RCITimerCallback	4-63
Hierarchy	4-63
Description	4-63
Enclosing Class	4-64
Constructor summary	4-64
Method summary	4-64
Constructor detail	4-64
Method detail	4-65
Class CRivDbEntity	4-65
Hierarchy	4-65
Description	4-65
Constructor summary	4-65
Method summary	4-66
Constructor detail	4-66
Method detail	4-66
Class CRivDummyCallback	4-67
Hierarchy	4-67
Description	4-68
Constructor summary	4-68
Method summary	4-68
Constructor detail	4-68
Method detail	4-69

Class CRivEvalClause	4-69
Hierarchy	4-69
Description	4-70
Constructor summary	4-70
Method summary	4-70
Constructor detail	4-71
Method detail	4-71
Class CRivException	4-72
Hierarchy	4-72
Description	4-72
All Implemented Interfaces	4-72
See Also	4-72
Field summary	4-73
Constructor summary	4-74
Method summary	4-74
Field detail	4-74
Constructor detail	4-80
Method detail	4-81
Inner Class CRivException.RivErrRec	4-82
Hierarchy	4-82
Description	4-82
Enclosing Class	4-82
Constructor summary	4-83
Method summary	4-83
Constructor detail	4-83
Method detail	4-84
Class CRivExpr	4-84
Hierarchy	4-84
Description	4-85
See Also	4-85
Field summary	4-85
Constructor summary	4-86
Method summary	4-86
Field detail	4-86
Constructor detail	4-87
Method detail	4-89

- Class CRivFilter **4-90**
 - Hierarchy **4-90**
 - Description **4-90**
 - See Also **4-90**
 - Constructor summary **4-91**
 - Method summary **4-91**
 - Constructor detail **4-92**
 - Method detail **4-93**
- Class CRivFilterParser **4-96**
 - Hierarchy **4-96**
 - Description **4-96**
 - See Also **4-96**
 - Constructor summary **4-97**
 - Method summary **4-97**
 - Constructor detail **4-97**
 - Method detail **4-98**
- Class CRivHashVector **4-100**
 - Hierarchy **4-100**
 - All Implemented Interfaces **4-100**
 - See Also **4-100**
 - Constructor summary **4-100**
 - Method summary **4-101**
 - Constructor detail **4-102**
 - Method detail **4-102**
- Class CRivMonitorFilterParser **4-107**
 - Description **4-107**
 - See Also **4-107**
 - Constructor summary **4-108**
 - Method summary **4-108**
 - Constructor detail **4-108**
 - Method detail **4-108**
- Class CRivQueryAtom **4-109**
 - Description **4-109**
 - See Also **4-109**
 - Constructor summary **4-109**
 - Constructor detail **4-111**
 - Method detail **4-112**

- Class CRivRecord **4-114**
 - Description **4-114**
 - See Also **4-115**
 - Constructor summary **4-115**
 - Method summary **4-115**
 - Constructor detail **4-116**
 - Method detail **4-117**
- Class CRivROMPer **4-122**
 - Hierarchy **4-122**
 - Description **4-123**
 - See Also **4-123**
 - Field summary **4-123**
 - Constructor summary **4-124**
 - Method summary **4-124**
 - Field detail **4-125**
 - Constructor detail **4-127**
 - Method detail **4-127**
- Class CRivRvDataHandler **4-135**
 - Description **4-135**
 - See Also **4-135**
 - Field summary **4-135**
 - Constructor summary **4-136**
 - Method summary **4-136**
 - Field detail **4-136**
 - Constructor detail **4-137**
 - Method detail **4-137**
- Class CRivTransport **4-141**
 - Description **4-141**
 - Field summary **4-141**
 - Constructor summary **4-142**
 - Method summary **4-142**
 - Field detail **4-143**
 - Constructor detail **4-145**
 - Method detail **4-145**

- Class CRivVarBind **4-150**
 - Description **4-150**
 - See Also **4-150**
 - Constructor summary **4-151**
 - Method summary **4-151**
 - Constructor detail **4-151**
 - Method detail **4-153**
- Class CRivVarBindList **4-154**
 - Hierarchy **4-154**
 - Description **4-155**
 - See Also **4-155**
 - Constructor summary **4-155**
 - Constructor detail **4-158**
 - Method detail **4-159**
- Interface IRivAlgebraic **4-175**
 - Description **4-175**
 - Hierarchy **4-175**
 - Field summary **4-175**
 - Field detail **4-175**
- Interface IRivConstants **4-176**
 - All known implementing classes **4-176**
 - Description **4-176**
 - Hierarchy **4-176**
 - See also **4-176**
 - Field summary **4-177**
 - Field detail **4-180**
- Interface IRivDataType **4-200**
 - All known implementing classes **4-200**
 - Description **4-200**
 - Hierarchy **4-200**
 - See Also **4-200**
 - Field summary **4-201**
 - Field detail **4-201**
- Interface IRivNodeType **4-203**
 - Description **4-203**
 - Hierarchy **4-203**
 - Field summary **4-203**
 - Field detail **4-203**

Interface IRivOper	4-204
All Known Implementing Classes	4-204
Description	4-204
Hierarchy	4-204
Field summary	4-205
Field detail	4-205
Interface IRivRecordListener	4-209
Description	4-209
Hierarchy	4-209
Method summary	4-209
Method detail	4-209
Interface IRivSubjects	4-210
Description	4-210
Hierarchy	4-210
Field summary	4-210
Field detail	4-211
Interface IRivTimerCallback	4-217
Description	4-217
Hierarchy	4-217
Method summary	4-217
Method detail	4-217
Class ParseException	4-218
Description	4-218
See Also	4-218
Field summary	4-219
Constructor summary	4-219
Field detail	4-219
Constructor detail	4-220
Method detail	4-222
Class RivFilter	4-222
Description	4-223
See Also	4-223
Field summary	4-223
Constructor summary	4-223
Method summary	4-223
Constructor Detail	4-225

- Interface RivFilterConstants **4-225**
 - Description **4-225**
 - All Known Implementing Classes: **4-225**
 - Field summary **4-226**
- Class RivFilterTokenManager **4-227**
 - Description **4-227**
 - See Also **4-227**
 - Field summary **4-228**
 - Constructor summary **4-228**
 - Method summary **4-228**
- Class RivMonitorFilter **4-228**
 - Description **4-229**
 - See Also **4-229**
 - Field summary **4-229**
 - Constructor summary **4-229**
 - Method summary **4-230**
 - Constructor detail **4-231**
- Interface RivMonitorFilterConstants **4-232**
 - Description **4-232**
 - Hierarchy **4-232**
 - All Known Implementing Classes **4-232**
 - See Also **4-232**
- Class RivMonitorFilterTokenManager **4-234**
 - Hierarchy **4-234**
 - Description **4-234**
 - See Also **4-234**
 - Field summary **4-235**
 - Constructor summary **4-235**
 - Method summary **4-235**
- Class Token **4-235**
 - Description **4-236**
 - See Also **4-236**
 - Field summary **4-236**
 - Constructor summary **4-236**
 - Method summary **4-236**
 - Field detail **4-237**
 - Constructor detail **4-238**
 - Method detail **4-239**

Class <code>TokenMgrError</code>	4-239
Description	4-240
See Also	4-240
Constructor summary	4-240
Method summary	4-240
Method detail	4-240
Class/Interface index	4-241
Class <code>ASCII_CharStream</code>	4-242
Class <code>CRivASN1Address</code>	4-243
Class <code>CRivAtom</code>	4-243
Class <code>CRivClient</code>	4-244
Class <code>CRivClientHelper</code>	4-244
Class <code>CRivClientInterface</code>	4-245
Class <code>CRivClientInterface.RCIRecordListener</code>	4-245
Class <code>CRivClientInterface.RCTimerCallback</code>	4-246
Class <code>CRivDbEntity</code>	4-246
Class <code>CRivDummyCallBack</code>	4-246
Class <code>CRivEvalClause</code>	4-246
Class <code>CRivException</code>	4-247
Class <code>CRivException.RivErrRec</code>	4-247
Class <code>CRivExpr</code>	4-248
Class <code>CRivFilter</code>	4-248
Class <code>CRivFilterParser</code>	4-248
Class <code>CRivHashVector</code>	4-249
Class <code>CRivMonitorFilterParser</code>	4-249
Class <code>CRivQueryAtom</code>	4-249
Class <code>CRivRecord</code>	4-250
Class <code>CRivROMPer</code>	4-250
Class <code>CRivRvDataHandler</code>	4-251
Class <code>CRivTransport</code>	4-251
Class <code>CRivVarBind</code>	4-252
Class <code>CRivVarBindList</code>	4-252
Interface <code>IRivAlgebraic</code>	4-253
Interface <code>IRivConstants</code>	4-254
Interface <code>IRivDataType</code>	4-255
Interface <code>IRivNodeType</code>	4-255
Interface <code>IRivOper</code>	4-256
Interface <code>IRivRecordListener</code>	4-256
Interface <code>IRivSubjects</code>	4-256
Interface <code>IRivTimerCallback</code>	4-257

Class *ParseException* 4-257
 Class *RivFilter* 4-257
 Interface *RivFilterConstants* 4-258
 Class *RivFilterTokenManager* 4-259
 Class *RivMonitorFilter* 4-259
 Interface *RivMonitorFilterConstants* 4-260
 Class *RivMonitorFilterTokenManager* 4-261
 Class *Token* 4-261
 Class *TokenMgrError* 4-261
 Summary 4-261

CHAPTER 5

Differences between Version 2 and NMOS Java API's 5-1

MWFM NMOS Java API replacements 5-1
 The *orv_web.kernel* package 5-2
 V2—*orv_web.kernel.OrvASN1Address* 5-2
 V2—*orv_web.kernel.OrvAtom* 5-3
 V2—*orv_web.kernel.OrvCryptograph* 5-5
 V2—*orv_web.kernel.OrvEvRec* 5-5
 V2—*orv_web.kernel.OrvModelInstance* 5-7
 V2—*orv_web.kernel.OrvPollDef* 5-8
 V2—*orv_web.kernel.OrvVarBind* 5-9
 The *orv_web.network* package 5-10
 V2—*orv_web.network.OrvClient* 5-10
 V2—*orv_web.network.OrvClientInterface* 5-11
 V2—*orv_web.network.OrvDataCallback* 5-14
 V2—*orv_web.network.OrvProfile* 5-15
 V2—*orv_web.network.OrvTransport* 5-17
 V2—*orv_web.network.OrvUser* 5-18
 The *orv_web.network.event* package 5-19



Introduction

This chapter provides an outline of the contents of the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide* documentation. An overview of each chapter is provided along with additional information on Cisco typographic conventions. Finally, we will discuss why you would use an API as well as the functionality of the NMOS Java API.

What is contained in this guide?

This guide contains a detailed description of the NMOS Java Application Programming Interface (API). The following table provides a brief summary of the contents of each chapter.

Table 1-1 Chapter Contents

Chapter	Description
Introduction	Illustrates the styles and conventions used in this document.
MWFM TIBCO/Rendezvous	Introduces the concepts behind Rendezvous and describes how it is utilized in the MWFM NMOS Java API.
Examples	Provides examples of some NMOS Java API programming.
The NMOS Java API Classes	Provides an overview of the NMOS Java API followed by a detailed list of all the published classes, interfaces, fields, constructors, and methods. Diagrams of class hierarchies are included.
Differences between Version 2 and NMOS Java API's	Details the differences in packages and classes between the version 2 Java API and the NMOS Java API. Classes which are superseded, or have become obsolete are highlighted.

Who is this guide for?

This guide is written for developers and technical services personnel who intend to write applications in Java that integrate with, or extend, the MWFM NMOS.

Readers should already be familiar with the Java programming language and Unix, as well as the material in the following documents:

- *Cisco Mobile Wireless Fault Mediator 2.0 - Fault Engineering Reference Guide*
- *Cisco Mobile Wireless Fault Mediator 2.0 - Topology and Platform Modeling Reference Guide*

Where should I start to read this book?

Chapter 2 describes the installation process and requirements and can be read independently for that purpose. Chapter 3 is a recommended starting point for new users of the MWFM NMOS Java API. Users that are familiar with MWFM and Java can refer to chapter 4 for details of classes.

Why use an Application Programming Interface?

An Application Programming Interface (API) is composed of the calls, subroutines, or software interrupts that enable programmers to build, customize, or link software applications. Its function is to enable a, usually, higher-level program to make use of, usually, lower-level services and functions of another software program. This particular API is used to interface to the core applications in MWFM and consists of a library of Java classes, fields, constructors, and methods that can be used for this purpose.

What are the capabilities of the NMOS Java API?

The NMOS Java API enables the user to interface into existing processes, but also to query any databases that have been set up. Note that in order to query a database, the process that you are trying to access (e.g. **riv_model**, **riv_class**), must be running.

For example, the NMOS Java API could be used in the following instances:

- To query **riv_model** to get the class name of a device that has generated an event.
- To select all entries in a certain table.
- To show all events with a severity equal to critical.
- To insert new entries into a table. For instance, if you wanted to add new classes to **riv_class**.
- To modify entries in databases, such as clearing events.
- To delete entries from databases. For instance, entities could be deleted from **riv_model**.
- To add listeners to listen for updates.

MWFM uses TIB®/Rendezvous™ for inter-process communications. This layer can be invisible to users of this API who only need to provide a domain name.

Styles used in this guide

Throughout the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide* we use certain specialized styles to highlight significant items. In the following sections, we describe how to interpret the meaning of those styles.

Note on command line examples

Command line examples in this guide use the ‘C’ shell environment.

What typographic conventions and symbols mean

Table 1-2 *Typographic conventions and symbols*

Typeface or symbol	Meaning	Example
Emphasis	Signals the class/interface/field/constructor/method being documented.	<code>public</code> <i>CRivASNIAddress</i> () .
Code	This typeface signals the names of commands, files and processes; On-screen computer output.	<code>\$ cd \$RIV_HOME</code>
File Name/Path	Denotes file and path names. Used throughout the installation chapter.	<code>\$RIV_HOME</code>

Summary

This chapter outlined the contents of the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide*. The next chapter will discuss areas of Rendezvous that are relevant to the MWFM NMOS API.



MWFM TIBCO/Rendezvous

This chapter will detail the aspects of Rendezvous that are important to users of the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide*. In particular, we will highlight the differences between `rvd` and `rva` that apply to Java and detail how you create a Rendezvous session.

How TIB/Rendezvous is utilized within MWFM

MWFM utilizes Rendezvous for inter-process communications. Encapsulated data is transferred across a network as self-describing messages, allowing for an efficient means of application-to-application broadcasting. The NMOS Java API also utilizes Rendezvous to query particular databases for relevant information. However, it should be noted that the classes in the NMOS Java API encapsulate the functionality provided by the Rendezvous classes, making this layer transparent to the user.

The Rendezvous process

The Rendezvous process that applies to Java is:

- **rvd**—The TIB/Rendezvous daemon. This is a background process through which all communications to, or from, a host machine must pass.
- **rva**—The TIB/Rendezvous agent. This is a background process which supports communication to, and from, Java applets.

When a Java application tries to initialize a new Rendezvous session, the session will try and connect to an **rvd** process that is already running. If such a process does not exist, one is automatically created.

Java applets cannot directly connect to a **rvd** process, but must connect to a **rva** process instead. The **rva** process, in turn, connects to a **rvd** process. For security reasons, remote Java applets cannot automatically start up a **rva** process if one is not already running, and must connect to an already established agent process.

How to create a new rvd session

A **rvd** session will try to connect to a **rvd** process with identical parameters, providing such a process has already been set up and is running. If no such process is available, the **rvd** session will start one on its local host computer and then connect to it.

A **rvd** session cannot automatically start processes on remote computers. If the parameters of a **rvd** session specify a remote daemon, then the daemon must already be running before the session attempts to connect.

Java programs are able to start **rvd** sessions on the local host computer. If a **rvd** session had to be started on a remote computer, this could be done by typing the following on the command line:

```
rvd
```

Initializing a rvd session via the NMOS Java API

The NMOS Java API provides the following `CRivClient` methods to initialize **rvd** sessions:

- `public int rcInitSession ()` – Initializes a **rvd** session for an independent Java application. It supplies `null` as the default for all three **rvd** parameters (`service`, `network`, and `daemon`) and will listen for application connections on TCP port 45001.
- `public int rcInitSession (String service, String network, String daemon)` – If a **rvd** process with these parameters already exists, the Java application connects to it automatically. `Service` specifies the service group to communicate with, `network` specifies the network to use for this session and `daemon` instructs the **rvd** session about how, and where, to find the Rendezvous daemon to establish communication with.

Please refer to `CRivClient` documentation for additional information.

How to create a new rva session

Java programs can never start **rva** sessions automatically as they must connect to an existing **rva** process.

The Rendezvous agent can be started using the **rva** command. The **rva** command line accepts **rvd** command parameters, and uses them to start **rvd**, when appropriate. To start a **rva** process, you can type the following on a command line, having previously killed off any existing **rvd** sessions:

```
rva -flavor 116
```

When **rva** starts, it attempts to connect to an **rvd** process with identical parameters. If an appropriate **rvd** process is not already running, **rva** starts it automatically. If **rvd** terminates, **rva** restarts it automatically. However, **rva** can start **rvd** only on the same computer as it cannot automatically start a remote **rvd**.

Numerous Java applets can connect to one **rva** process, and each applet can create several `RvSession` connections to **rva**. A **rva** process instance connects to **rvd** only once, with a single session, regardless of the number of Java sessions that connect to it.

Initializing a rva session via the NMOS Java API

The NMOS Java API provides the following `CRivClient` methods to initialize **rva** sessions:

- `public int rcInitSession(String hostname)`– Attempts to connect to a rva session for Java applets. TCP port 7600 is used as the default port. The argument `hostname` specifies the Rendezvous agent (**rva**) to connect to that is running on the computer with this hostname.
- `public int rcInitSession(String hostname, int port)`– Attempts to connect to a rva session for Java applets. The Rendezvous agent (**rva**) will connect to the TCP port specified in the arguments.

Please refer to the `CRivClient` documentation for additional information.

Summary

This chapter has provided a brief overview of how Rendezvous is utilized within MWFM. We have also detailed how new Rendezvous sessions are created and the differences between **rva** and **rva** that apply to Java. The next chapter will work through some examples of the NMOS Java API functionality as well as the test application that is provided.



Examples

This chapter will work through some NMOS Java API programming examples. The following examples will enable you to: start a Rendezvous `rvd` session, obtain class records stored in `riv_class`, query `riv_model`, insert and modify entries into `riv_model`, delete an entry from `riv_model`, and add and remove a listener for updates from `riv_class`. The `CRivClientHelper` test applications have also been provided, which enable you to, amongst other things, retrieve records from `riv_auth`, and add new users into `riv_auth`.

The NMOS Java API test application

The NMOS Java API includes a Test Application called **TestApp.java**. The examples from this chapter, as well as the screen shots, are taken from this test application.

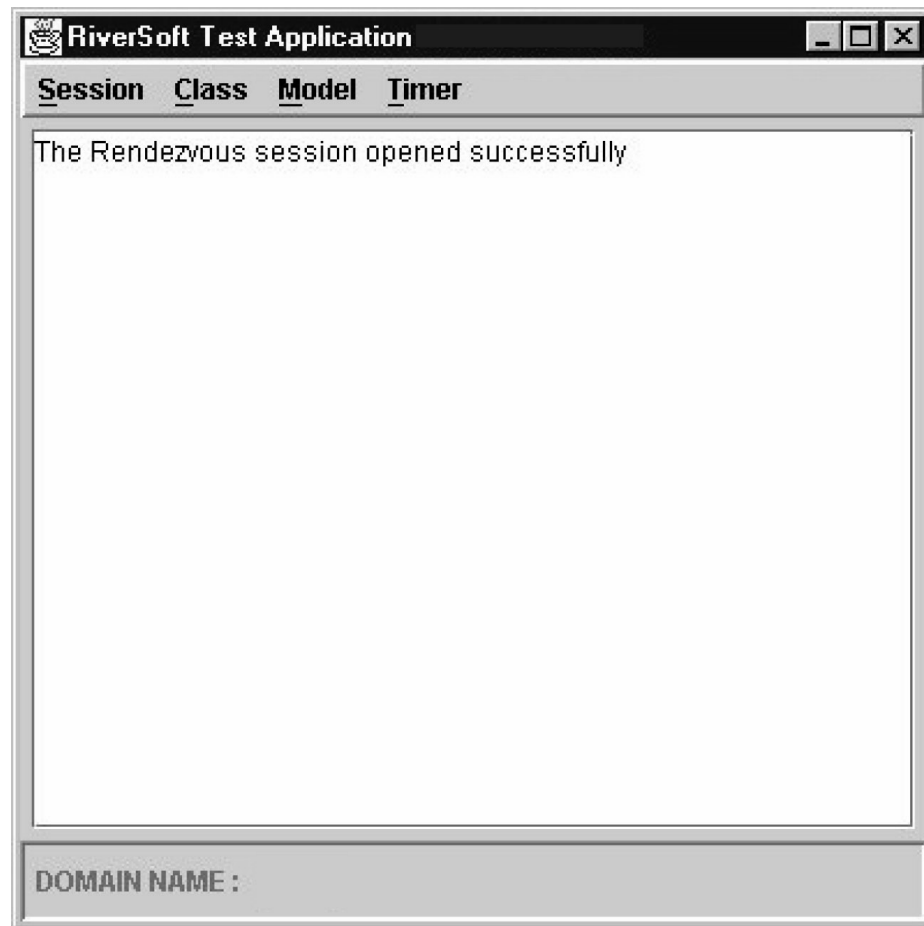
Starting a Rendezvous `rvd` session

This extract provides an example of how to open a new Rendezvous session to query a MWFM domain. Connect to a new Rendezvous `rvd` session using the `rcInitSession()` method. If you want to connect to a `rvd` session that was already running, use the `rcInitSession(String service, String network, String daemon)` method.

If successful, the message “The Rendezvous session opened successfully” will appear in the application window, and enables the items on the data menu.

Once you have successfully opened a Rendezvous session, you will see the following:

```
1  public void initialiseClient()
2  {
3  ! Make the CRivClient object, with the specified domain name and
   latency.
   m_Client = new CRivClient(CRivClient.RIV_DEFAULT_LATENCY,
   m_DomainName);
4  ! Now try and initialise a rvd session. Check the return value to
   see if it opened successfully.
   int opened = m_Client.rcInitSession();
5  ! 5. Make the CRivClientinterface object—this is used as a
   convient way to interface with the client.
   m_ClientInterface = new CRivClientInterface(m_Client);
6  if (opened == CRivException.RIV_OK)
7  {
8  m_ProgressTextArea.setText("The Rendezvous session opened" +
   "successfully. ");
9  ! 9-14. Report if successful.
   {
10 else
11 {
12 m_ProgressTextArea.setText("Could not open Rendezvous" +
   "session. ");
13 }
14 }
```

Obtaining all class records stored in riv_class

This extract provides an example of how to show all the entries in a **riv_class** database. The results are displayed in the application window. The query is done via the `CRivClientInterface` object.

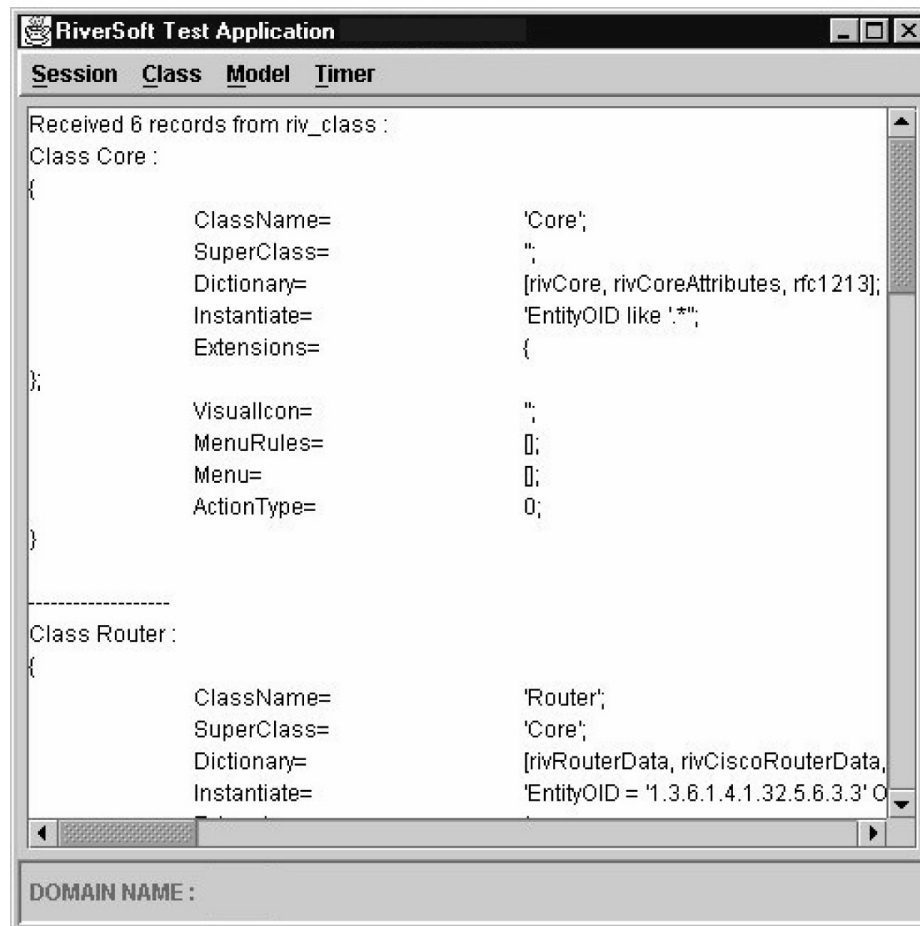
```

1  public void showAllClasses()
2  {
3      if (m_Client != null)
4      {
5          ! Create an instance of the inner class RivClassListener (see
           below) so the results from the query can be displayed.
           RivClassListener classListener = new RivClassListener();
6          ! Make a query on the Cisco class Query subject which is:
           "IRivSubjects.RIV_CLASS_QUERY_SUBJ"
           String qrySubject = IRivSubjects.RIV_CLASS_QUERY_SUBJ;
7          ! Construct the OQL query String to obtain all the classes
           currently held in riv_class for this domain.
           String qryString = "select * from class.activeClasses;";
8          ! 8-18. Spend the query to the Client. There is no need to create
           CRivDataHandlers, or data processing threads—this is all
           handed by the CRivClient interface object.
           int rc = m_ClientInterface.rciTimedSendToClient( qrySubject,
               qryString, m_Client.getRCTimeOut(), classListener);
9          if (rc == CRivException.RIV_OK)
10         {
11             m_ProgressTextArea.setText("Awaiting results from" +
               "riv_class...");
12         }
13         else
14         {
15             m_ProgressTextArea.setText("Failed to send query to" +
               "riv_class.");
16         }
17     }
18 }
19 ! 19-20. An inner class to listen for CRivRecords arriving for
           queries from riv_class.
           class RivClassListener implements IRivRecordListener
20 {

```

```
21 ! 21-29. Method will be called when a list of CRivRecords is process
    from the transport layer.
    public void rrlRivRecordsReceived(CRivRecord[] rivRecs)
22     {
23         m_ProgressTextArea.setText("Received " + rivRecs.length +
            "records from riv_class :\n");
24         if (rivRecs != null)
25         {
26             int i;
27             for (i = 0; i < rivRecs.length; i++)
28             {
29                 CRivRecord classRec = rivRecs[i];
30                 CRivAtom className =
                    classRec.getRRValueOf ("ClassName");
31                 m_ProgressTextArea.append("Class " + className +
                    ":\n");
32                 m_ProgressTextArea.append(classRec.toString());
33                 m_ProgressTextArea.append("\n-----\n");
34             }
35         }
36     }
37 ! 37-41. Method will be called when the time limit has expired for
    records to be received through the Transport Layer from Rendezvous.
    public void rrlTimeOutReceived()
38     {
39         m_ProgressTextArea.setText("Timed out on response from" +
            "riv_class.");
40     }
41 }
```

The format of the received records will look like this:



Querying riv_model

This extract provides an example of how to show all the entries in a **riv_model** database.

The results are displayed in the application window. This method does not use the `CRivClientInterface` object, but uses the `CRivClient` directly.

```

1  public void showAllModelData()
2  {
3      if (m_Client != null)
4      {
5          ! Create an instance of the inner class RivModelListener (see
          below) so the results from the query can be displayed.\

          RivModelListener modelListener = new RivModelListener();

```

```

6      ! 6-10. Make a CRivRvDataHandler to handle queries to riv_model
      if necessary

          if (m_ModelQryHandler == null)
7          {
8              m_ModelQryHandler = new CRivRvDataHandler
              (modelListener);
9          }

10         modelListener.setRvDataHandler(m_ModelQryHandler);

11      ! 11. Make a query on the Cisco Model Query subject which is:
      "IRivSubjects.RIV_MODEL_QUERY_SUBJ".

          String qrySubject = IRivSubjects.RIV_MODEL_QUERY_SUBJ;

12      ! 12. Construct the OQL query String to obtain all the entities
      currently held in riv_model for this domain. If we only wanted to
      obtain the model records which have a Description field of
      "Switch" we could use an OQL query String of: "select * from
      master.entityByName" where Description="Switch".

          String qryString = "select * from" +
              "master.entityByName;";

13      ! 13-15. Send the query to the Client.

          int rc = m_Client.rcTimedSend(qrySubject, qryString,
              m_Client.getRCTimeOut(), m_ModelQryHandler);

14          if (rc == CRivException.RIV_OK)
15          {

16      ! 16-24. Kick off data processing in our CRivRvDataHandler.

          m_ModelQryHandler.rrdhStartThread();

17          m_ProgressTextArea.setText("Awaiting results from" +
              "riv_model...");

18          }

19          else

20          {

21          m_ProgressTextArea.setText("Failed to send query" +
              "to riv_model.");

22          }

23          }

24      }

25      ! 25-27. An inner class to listen for CRivRecords arriving for
      queries/updates from riv_model.

      class RivModelListener implements IRivRecordListener

26      {

```

```

27     private CRivRvDataHandler m_RvHandler;
28     ! 28-31. Set the CRivRvDataHandler which will pass the records
      onto this listener.
      public void setRvDataHandler(CRivRvDataHandler rvCallback)
29     {
30         m_RvHandler = rvCallback;
31     }
32     ! 32-46. Method will be called when a list of CRivRecords is
      processed from the transport layer
      public void rrlRivRecordsReceived(CRivRecord[] rivRecs)
33     {
34         if (rivRecs != null)
35         {
36             if (rivRecs.length == 1)
37             {
38                 m_ProgressTextArea.setText("Received a record from" +
39                 "riv_model :\n");
39             }
40             else
41             {
42                 m_ProgressTextArea.setText("Received rivRecs.length" +
43                 "records from riv_model:\n");
43             }
44             int i;
45             for (i = 0; i < rivRecs.length; i++)
46             {
47                 ! 47-55. Display the EntityName field, followed by the whole
38                 CRivRecord.
39                 CRivRecord modelRec = rivRecs[i];
40                 CRivAtom entityName = modelRec.getRRValueOf
41                 ("EntityName");
42                 m_ProgressTextArea.append("EntityName" + entityName +
43                 ":\n");
44                 m_ProgressTextArea.append(modelRec.toString());
45                 m_ProgressTextArea.append("\n-----\n");
46             }
47         }
48     }
49     if ( (m_ModelQryHandler != null) &&
50         (m_RvHandler == m_ModelQryHandler) )

```

```

55     {
56     ! 56-59. The query is finished-stop the data Threads.
           m_RvHandler.rrdhCancelListener();
57         m_RvHandler.rrdhStopThread();
58     }
59 }

```

The records received will be in the following format:



Inserting an entry into riv_model

This extract provides an example of how to insert an example MODEL record into an `EntityByName` table in a master database in `riv_model`. Please note that in this example, the master database is being populated from scratch and therefore there will be no entries before these are added.

For more information on the `EntityByName` table, please refer to Chapter 4, “The NMOS Java API Classes”.



Note

Construct the OQL insert String to insert a new entry, with an `EntityName` of “javatest” into the `master.entityByName` table. These entries correspond to those set out in `ModelSchema.cg`.

```

1  StringBuffer insertBuffer = new StringBuffer();
2
3  insertBuffer.append("insert into master.entityByName values (");
4  ! ObjectId - insert as 0, filled out by riv_model.
   insertBuffer.append("0,");
5  ! EntityName
   insertBuffer.append("\"javatest\",");
6  ! Address
   insertBuffer.append("[\"\", \"\", \"194.203.200.11\", \"\", \"\", \"\"]");
7  ! Description
   insertBuffer.append("\"SunOS otho5.6 Generic sun4u\",");
8  ! EntityType
   insertBuffer.append("2,");
9  ! ClassName
   insertBuffer.append("\"Core\",");
10 ! EntityOID
   insertBuffer.append("\"1.3.6.1.4.1.2021.250.3\",");
11 ! Status
   insertBuffer.append("0,");
12 ! Security
   insertBuffer.append("\"public\",");
13 ! RelatedTo
   insertBuffer.append("[\"claudius\",");
14 ! Contains
   insertBuffer.append("[],");
15 ! PartOfName
   insertBuffer.append("[],");

```



```

16  ! IsActive
    insertBuffer.append("1,");

17  ! CreateTime
    insertBuffer.append("964528083,");

18  ! ChangeTime
    insertBuffer.append("964528083,");

19  ! ActionType
    insertBuffer.append("0;");

20  String insertString = insertBuffer.toString();

```

Modifying an entry in riv_model

This extract shows how to modify the example MODEL record in the `EntityByName` table in the master database in `riv_model`.

```

1  public void modifyTestModelRecord()
2  {
3      if (m_Client != null)
4      {
5          ! Send the modify on the RiverSoft Model Query subject, which is:
          "IRivSubjects.RIV.MODEL_QUERY_SUBJ".
          String qrySubject = IRivSubjects.RIV_MODEL_QUERY_SUBJ;

6          ! 6-10. Construct the OQL update String to update the entry with an
          EntityName of "javatest" in the master.entityByName table. Set the
          classname to be Router.
          StringBuffer updateBuffer = new StringBuffer();

7          updateBuffer.append("update master.entityByName ");
8          updateBuffer.append("set ClassName=\"Router\"");
9          updateBuffer.append(" where EntityName=\"javatest\"");

10         String updateString = updateBuffer.toString();

11         ! Create an empty CRivDummyCallback () - the form of the reply from
          the insert is not important.
          int rc = m_Client.rcSend(qrySubject, updateString, new
          CRivDummyCallback());

12         ! Display whether the modification was successfully sent to
          riv_model.
          if (rc == CRivException.RIV_OK)

13             {

14                 m_ProgressTextArea.setText("Sent update to" +
          "master.entityByName in riv_model.");

15             }

```

```

16     else
17     {
18         m_ProgressTextArea.setText("Failed to update record in"
19         + "riv_model.");
19     }
20 }
21 else
22 {
23     m_ProgressTextArea.setText("Failed to update record in" +
24     "riv_model.");
24 }
25 }

```

Deleting an entry from riv_model

This extract shows how to delete the example MODEL record from the `EntityByName` table in the master database in `riv_model`.

For information on the `EntityByName` table, please refer to Chapter 4, “The NMOS Java API Classes”.

```

1  public void deleteTestModelRecord()
2  {
3      if (m_Client != null)
4      {
5          ! Send the modify on the RiverSoft Model Query subject which is:
          "IRivSubjects.RIV.MODEL_QUERY_SUBJ".
          String qrySubject = IRivSubjects.RIV_MODEL_QUERY_SUBJ;
6          ! 6-9. Construct the OQL delete String to delete the entry with an
          EntityName of "javatest" from the master.entityByName table.
          StringBuffer deleteBuffer = new StringBuffer();
7          deleteBuffer.append("delete from master.entityByName ");
8          deleteBuffer.append(" where EntityName=\"javatest\"");
9          String deleteString = deleteBuffer.toString();
10         ! Create an empty CRivDummyCallback() - the form of the reply from
          the delete is not important.
          int rc = m_Client.rcSend(qrySubject, deleteString, new
          CRivDummyCallback());
11         ! 11-24. Display whether the delete was successfully sent to
          riv_model.
          if (rc == CRivException.RIV_OK)
12         {

```

```

13         m_ProgressTextArea.setText("Sent delete to" +
14             "master.entityByName in riv_model.");
15     }
16     else
17     {
18         m_ProgressTextArea.setText("Failed to delete record
19             from" + "riv_model.");
20     }
21     else
22     {
23         m_ProgressTextArea.setText("Failed to delete record from" +
24             "riv_model.");
25     }

```

Adding and removing a listener for updates from riv_class

This extract shows how to add, or remove, a listener for updates in `riv_class`. The records are displayed in the application window. The application will toggle between two states to add or remove the listener.

```

1     public void addRemoveClassListener()
2     {
3         if (m_ClassUpdateListener == null)
4         {
5             ! Add a listener for updates in riv_class. The inner class
6             RivClassListener implements the IRivRecordListener interface to
7             display the CRivRecords which arrive as updates from riv_class.
8             m_ClassUpdateListener = new RivClassListener();
9
10            ! 6-21. Tell the client to add on a listener for class updates, the
11            Class notify subject for which is
12            "IRivSubjects.RIV.CLASS_NOTIFY_SUBJ". Use the
13            CRivClientInterface object again for convenience
14
15            int doneOK = m_ClientInterface.rciAddService
16                (IRivSubjects.RIV_CLASS_NOTIFY_SUBJ, m_ClassUpdateListener);
17
18            if(doneOK == CRivException.RIV_OK)
19            {
20                m_ProgressTextArea.setText("Listener added to display" +
21                    "riv_class updates.");
22            }
23        }
24        else

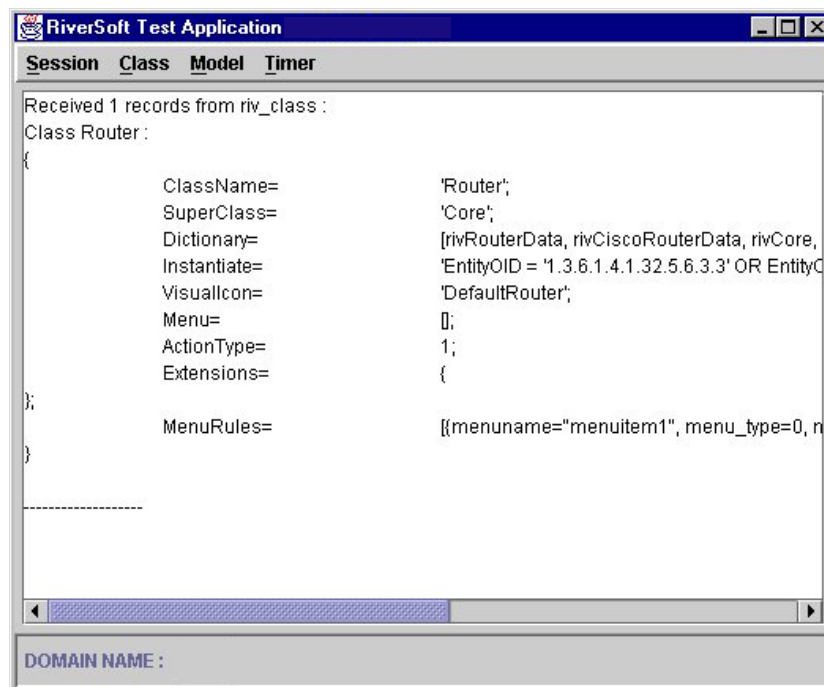
```

```

12     {
13         CRivException err = new CRivException(doneOK, "Failed to"
14         + "enable model updates");
15         err.rePrintErrorMessage(true);
16         m_ProgressTextArea.setText("Failed to add listener to" +
17         "display riv_class updates.");
18         m_ClassUpdateListener = null;
19     }
20 }
21 else
22 {
23     m_ProgressTextArea.setText("Request made to stop displaying" +
24     "riv_class updates.");
25     m_ClientInterface.rciRemoveService (m_ClassUpdateListener);
26     m_ClassUpdateListener = null;
27 }
28 }

```

Once an update has been received, it will appear in the following format:



The NMOS Java API CRivClientHelper test applications

Initializing the application, the Rendezvous session and the CRivClientHelper object

The following examples have been written so that they work as stand-alone applications. Therefore, each of the examples will need to begin with the following lines of code. This is necessary to initialize the application, the Rendezvous session and the CRivClientHelper object.

```
1  import com.Cisco .riv_web.vertigo.*;
2
3  public class ClientHelperTest
4  {
5
6      public static void main(String args[])
7      {
8          String domain = null;
9
10         if(args.length < 1)
11         {
12             domain = "JAVAAPI";
13         }
14         else
15         {
16             domain = args[0];
17         }
18
19         new ClientHelperTest(domain);
20     }
21
22     public ClientHelperTest(String m_DomainName)
23     {
24         CRivClient m_Client = null;
25         CRivClientHelper helper = null;
26
27         ! 27-32. Set up the session.
28
29         m_Client = new CRivClient(CRivClient.RIV_DEFAULT_LATENCY,
30                                 m_DomainName);
```

```

28     int opened = m_Client.rcInitSession();
29     if( CRivException.RIV_OK == opened )
30     {
31         System.out.println("Session opened successfully");
32     }
33
34     ! Construct a new client helper object ensuring that a correct
      username and password are supplied.
      CRivClientHelper clientHelper = null;
35
36     String username = "TestUser";
37     String password = "12345";
38
39     try
40     {
41         clientHelper = new CRivClientHelper(m_Client, username,
      password);
42     }
43     catch (CRivException e)
44     {
45         System.out.println("Exception in creating
      CRivClientHelper :");
46         System.out.println(e.getMessage());
47         System.exit(0);
48     }

```

Obtaining all event records from riv_f_amos

```

1     try
2     {
3         CRivRecord[] eventRecs =
      clientHelper.getRCHallEventRecords();
4
5     Display the event records.
      System.out.println("\nNumber of event records is " +
      eventRecs.length);

```

```
6
7     int i = 0
8     for (i = 0; i < eventRecs.length; i++)
9     {
10        System.out.println(eventRecs[i].toString());
11    }
12 }
13 catch (CRivException re)
14 {
15     System.out.println("\nException getting event records : " +
16         re);
17 }
```

Obtaining event records from riv_f_amos that match a list of criteria

In this example, we will obtain all events of type Alert, with a ClassName of Router.

```
1     try
2     {
3         ! 3-8. Construct the CRivVarBinds for the query.
4         CRivVarBind vb1 = new CRivVarBind(new CRivAtom("EventType"),
5             new CRivAtom(IRivConstants.REC.Alert));
6         CRivVarBind vb2 = new CRivVarBind(new
7             CRivAtom("ClassName"), new CRivAtom("Router"));
8         CRivVarBindList vbList = new CRivVarBindList();
9         vbList.rvblAdd(vb1);
10        vbList.rvblAdd(vb2);
11        CRivRecord[] eventRecs =
12            clientHelper.getRCHEventRecords(vbList);
13
14        ! Display the event records.
15        System.out.println("\nNumber of event records is " +
16            eventRecs.length);
17
18        int i = 0;
19        for (i = 0; i < eventRecs.length; i++)
20        {
21            System.out.println(eventRecs[i].toString());
22        }
23    }
```

```

17     }
18     catch (CRivException re)
19     {
20         System.out.println("\nException getting event records : " +
21             re);
21     }

```

Clearing an event from riv_f_amos

In this example, we will try to clear an event from `riv_f_amos` with `EventId = 31`.

```

1  ! 1-2. Create CRivRecord to represent the event we want to delete.
   The only field it needs to contain is EventId.
   ! To verify that this event has been cleared, repeat the steps in the
   "Obtaining all event records from riv_f_amos" example.
   CRivRecord eventToClear = new CRivRecord();
2  eventToClear.rrAddValue(
   new CRivAtom(IRivConstants.RIV_AMOS_FN_EVID),
   new CRivAtom(31));
3
4  try
5  {
6      clientHelper.rchClearEventRecord(eventToClear);
7  }
8  catch (CRivException re)
9  {
10     System.out.println("\nException clearing event record : " +
11         re);
11 }

```

Obtaining all users from riv_auth

```

1  try
2  {
3      CRivRecord[] userRecs = clientHelper.getRCHAuthUsers();
4
5  ! Display the user records.
   System.out.println("\nNumber of user records is " +
6      userRecs.length);

```



```
6
7     int i = 0;
8     for (i =0; i < userRecs.length; i++)
9     {
10        System.out.println(userRecs[i].toString());
11    }
12 }
13 catch (CRivException re)
14 {
15     System.out.println("\nException getting users held in" +
16         "riv_auth : " + re);
17 }
18 ! 18-20. Get all the profile records from riv_auth.
19     try
20     {
21         CRivRecord[] profileRecs =
22             clientHelper.getRCHAuthProfiles();
23     }
24     ! Display the profile records.
25     System.out.println("\nNumber of profile records is " +
26         profileRecs.length);
27     for (int i = 0; i < profileRecs.length; i++)
28     {
29         System.out.println(profileRecs[i].toString());
30     }
31 }
32 catch (CRivException re)
33 {
34     System.out.println("\nException getting profiles held in" +
35         "riv_auth : " + re);
36 }
```

Inserting a new user into riv_auth

Each user must have a Name, Password, and a Profile.

```
1    ! Create a CRivVarBind for each of the new user's name, password, and
      profile name.
      CRivVarBind nameVb = new CRivVarBind(new CRivAtom("Name"),
      new CRivAtom("Reader"));
2    CRivVarBind passVb = new CRivVarBind(new CRivAtom("Password"),
      new CRivAtom("123"));
3    CRivVarBind profVb = new CRivVarBind(new CRivAtom("Profile"),
      new CRivAtom("readonly"));
4
5    ! 5-8. Put the CRivVarBinds into a CRivVarBindList, and use this to
      create a new CRivRecord of the new user.
      CRivVarBindList insertList = new CRivVarBindList();
6    insertList.rvblAdd(nameVb);
7    insertList.rvblAdd(passVb);
8    insertList.rvblAdd(profVb);
9
10   CRivRecord newUser = new CRivRecord(insertList);
11
12   try
13   {
14   ! Insert the new user record into riv_auth.
      boolean sentInsert =
          clientHelper.rchInsertUserRecord(newUser);
15   System.out.println("Request sent to insert new user : ");
16   System.out.println(newUser.toString());
17   }
18   catch (CRivException re)
19   {
20   System.out.println("\nException sending new user request" +
      "to riv_auth : " + re);
21   }
```

Inserting a new user profile into riv_auth

Each Profile must have a Name, Rank, and a Permissions list. The Permissions list records whether the Profile allows a user to perform various actions in different categories, such as Clear an event, or Create a User.

```
22  ! 22-23. Create a CRivVarBind for each of the new Profile's name,
      rank, and permissions-the Profile name and
      Rank are straightforward.
      CRivVarBind profNameVb = new CRivVarBind(new
          CRivAtom("Profile"), new CRivAtom("APITest"));
23      CRivVarBind rankVb = new CRivVarBind(new CRivAtom("Rank"),
          new CRivAtom(15));
24
25  ! 25-33. Construct a CRivVarBindList of the user's permissions.
      CRivVarBind usersView = new CRivVarBind("pView", 1);
26      CRivVarBind usersChange = new CRivVarBind("pChange", 1);
27      CRivVarBind usersCreate = new CRivVarBind("pCreate", 0);
28      CRivVarBind usersDel = new CRivVarBind("pDelete", 0);
29      CRivVarBindList usersList = new CRivVarBindList(4);
30      usersList.rvblAdd(usersView);
31      usersList.rvblAdd(usersChange);
32      usersList.rvblAdd(usersCreate);
33      usersList.rvblAdd(usersDel);
34
35  ! 35-36. Put this CRiVarBindList in a CRivAtom, to form the User={}
      CRivVarBind.
      CRivAtom usersAtm = new CRivAtom(usersList);
36      CRivVarBind usersVb = new CRivVarBind("Users", usersAtm);
37
38  ! 38-46. Construct a CRiVarBindList of the event permissions.
      CRivVarBind evAss = new CRivVarBind("pAssign", 1);
39      CRivVarBind evAck = new CRivVarBind("pAck", 1);
40      CRivVarBind evDeAck = new CRivVarBind("pDeAck", 0);
41      CRivVarBind evClr = new CRivVarBind("pClear", 0);
42      CRivVarBindList eventsList = new CRivVarBindList(4);
43      eventsList.rvblAdd(evAss);
44      eventsList.rvblAdd(evAck);
45      eventsList.rvblAdd(evDeAck);
46      eventsList.rvblAdd(evClr);
```

```

47
48 ! 48-49. Put this CRivVarBindList in a CRivAtom, to form the Event={ }
   CRivVarBind.
       CRivAtom eventsAtm = new CRivAtom(eventsList);
49     CRivVarBind eventsVb = new CRivVarBind("Events", eventsAtm);
50
51 ! 51-61. Put these two CRivVarBind into new OBJECT type CRivAtoms,
   and add these atoms to the Permissions vector.
       CRivVarBindList userPermList = new CRivVarBindList();
52     userPermList.rvblAdd(usersVb);
53     CRivAtom userPermAtm = new CRivAtom(userPermList);
54
55     CRivVarBindList eventPermList = new CRivVarBindList();
56     eventPermList.rvblAdd(eventsVb);
57     CRivAtom eventPermAtm = new CRivAtom(eventPermList);
58
59     java.util.Vector permList = new java.util.Vector();
60     permList.addElement(userPermAtm);
61     permList.addElement(eventPermAtm);
62
63 ! Create the overall Permissions CRivVarBind for the new Profile
   record.
       CRivVarBind permsVb = new CRivVarBind(
           new CRivAtom("Permissions"),
           new CRivAtom(permList));
64
65 ! 65-68. Add this Permissions CRivVarBind to a list along with the
   Profile name and Rank CRivVarBind we created earlier.
       CRivVarBindList profInsertList = new CRivVarBindList();
66     profInsertList.rvblAdd(profNameVb);
67     profInsertList.rvblAdd(rankVb);
68     profInsertList.rvblAdd(permsVb);
69
70 ! Create the new Profile CRivRecord and insert the new Profile into
   riv_auth.
       CRivRecord newProfile = new CRivRecord(profInsertList);
71
72     try
73     {

```

```
74         boolean sentProfInsert =
              clientHelper.rchInsertProfileRecord(newProfile);
75
76         System.out.println("Request sent to insert new profile : ");
77         System.out.println(newProfile.toString());
78     }
79     catch (CRivException re)
80     {
81         System.out.println("Exception sending new profile" +
              "request to riv_auth : " + re);
82     }
```

Modifying an existing profile held in riv_auth

In this example, we will change the user's rank to 40.

```
1     ! 1-4. Create a list of the modifications to make the Profile-in this
              case it is a list of just one CRivVarBind.
              CRivVarBind rankChange = new CRivVarBind(
                  newCRivAtom(IRivConstants.RIV_AUTH_DB_RANK),
                  new CRivAtom(40));
2
3     CRivVarBindList modsList = new CRivVarBindList();
4     modsList.rvblAdd(rankChange);
5
6     ! 6-10. The profile we want to change is the APITest on we created
              earlier.
              ! Create a CRivRecord of this profile to change-only the Profile
              field needs to be set.
              CRivVarBind profileVb = new CRivVarBind(
                  new CRivAtom("Profile"),
                  new CRivAtom("APITest"));
7     CRivRecord profileToUpdate = new CRivRecord();
8     profileToUpdate.rrAddValue(profileVb);
9
10    clientHelper.rchUpdateProfileRecord(profileToUpdate,
              modsList);
```

Determining if a user has permission to clear an event

Use the `getRCHAuthPermission` (String category, String action) method, passing in the Events category and clearing action Strings from `IRivConstants`.

```

1    ! 1-2. Finish up by closing the client session.
      boolean eventsClear = clientHelper.getRCHAuthPermission(
          IRivConstants.RIV_AUTH_DB_EVENTS_OBJECT,
          IRivConstants.RIV_AUTH_FIELD_EV_CLEAR);
2
3    System.out.println("Does user " + username + " have " +
      " permission to " + " clear events : " + eventsClear);

```

Closing the session

Each of the `CRivClientHelper` applications must end with the following code, in order to close the client session.

```

1    int closed = m_Client.rcTerminateSession();
2    if(closed == CRivException.RIV_OK)
3    {
4        System.out.println("Session closed successfully");
5    }
6    else
7    {
8        System.out.println("Session could not be closed");
9    }
10
11 }
12 }

```

Summary

This chapter has detailed some examples of Java programming that will enable you to: start a Rendezvous `rvd` session; obtain class records stored in `CLASS`; query `riv_model`; insert, modify, and delete entries into `riv_model`; add and remove a listener for updates from `riv_class`; obtain event records from `riv_f_amos`; clear events from `riv_f_amos`; obtain all the users from `riv_auth`; modify an existing profile in `riv_auth`; determine if a user has permission to clear an event. The next chapter provides a full reference listing of all the classes and interfaces associated with the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide*.



The NMOS Java API Classes

This chapter will detail and describe the Java package, classes, and interfaces published with the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide*. Each class and interface is documented in alphabetical order along with its associated fields, constructors, and methods.

Chapter organization

First, the hierarchy of the classes within the `com.riversoft.riv_web.vertigo` package is depicted in the form of a graphic. Then a summary of all the classes and interfaces is provided along with a brief description of their functionality. Next, we list the reference pages for each class and interface, in alphabetical order.

Class Java—The name of the class.

`(com.riversoft.riv_web.vertigo)` - The package to which this class belongs.

Hierarchy

A graphic representing how this class fits into the package hierarchy.

Description

A full description of this class.

See Also

Other classes, or interfaces, that the reader should be aware of when trying to implement areas of this class' functionality.

Field/Constructor/Method Summary

A summary of the fields/constructors/methods of this class, with the relevant reference pages.

Field/Constructor/Method Detail

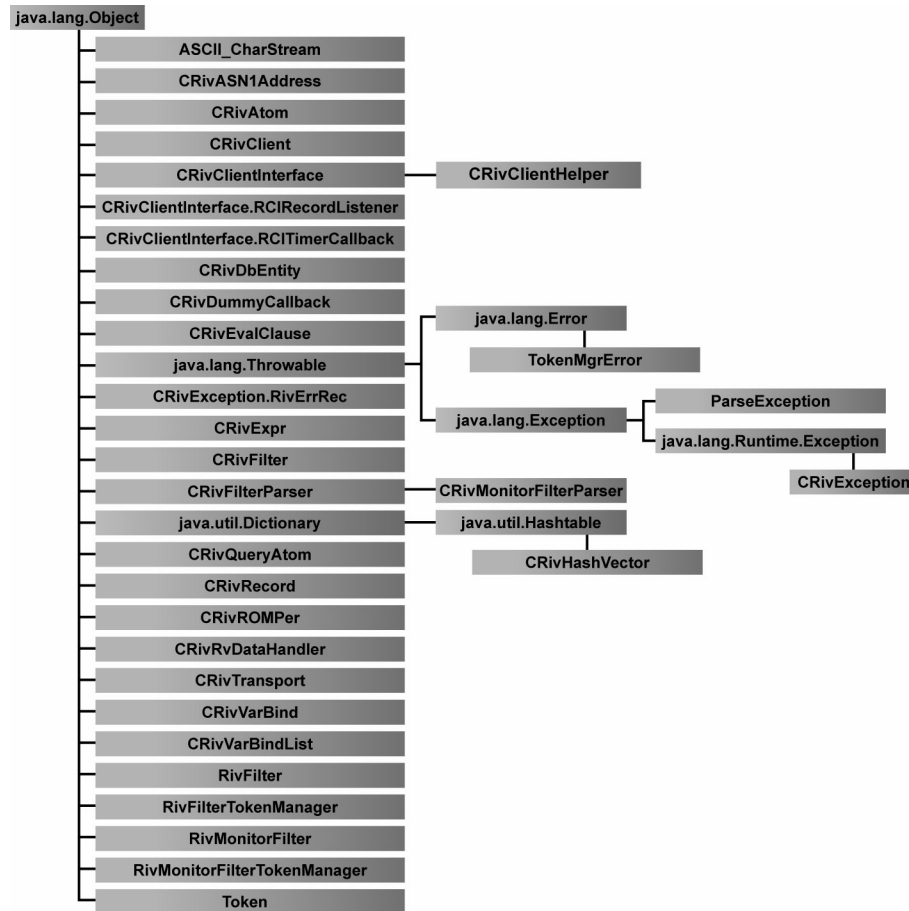
Full documentation of the fields/constructors/methods. For information on the layout of the documentation, please see the table below:

Field/Constructor/Method	The name of the field, constructor, or method that is being documented.
Description	A description of the field, constructor, or method.
Parameters	A description of the input parameters, if there are any.
Returns	A description of the return value, if any. Where it is obvious from the description what the return value would be, no separate listing is given.
Throws	The exception that is thrown when this method encounters an error.
Specified By	For a method, in a class which is implemented from an interface, the name of the interface which specifies that method is stated here.

Index

An index, for easy reference, listing all NMOS Java API classes and interfaces, with associated fields, constructors and methods.

The `com.riversoft.riv_web.vertigo` package. In the MWFM NMOS Java API all classes are encompassed by the one package, `com.riversoft.riv_web.vertigo`. Figure 4-1 illustrates this.

Figure 4-1 Hierarchy of the *com.riversoft.riv_web.vertigo* package

Summary of classes

The table below summarizes the functionality of each class within the NMOS Java API. Please note that some of the classes use the username and password.

Class	Description
1. <i>ASCII_CharStream</i>	An input stream class used by parsers, where the stream is assumed to contain only ASCII characters (without unicode processing).
2. <i>CRivASN1Address</i>	The class <i>CRivASN1Address</i> contains hierarchical style addressing such as 1.2.3.4 and is used to hang both the topology model and class model inside of all engines that need it (particularly the riv_disco engine).
3. <i>CRivAtom</i>	This class implements the storage of the fundamental types, such as <i>String</i> , <i>int</i> , and <i>asn1</i> idents, as well as providing methods for comparisons between them.

Class	Description
4. <i>CRivClient</i>	This class manage sessions to Rendezvous, handles initialization and client handling.
5. <i>CRivClientHelper</i>	This class is an extension of <i>CRivClientInterface</i> and acts as an intermediary between a <i>CRivClient</i> object and user defined classes. It provides convenience methods to allow polling of an active MWFM domain and abstracts out the Rendezvous methods and objects.
6. <i>CRivClientInterface</i>	This class acts as an interface to a given <i>CRivClient</i> , providing convenience methods to allow polling to an active domain.
7. <i>CRivClientInterface.RCIRecordListener</i>	This is an inner class implementing the <i>IRivRecordListener</i> interface which listens for replies on queries or listener activities from the <i>CRivClient</i> . This acts as an intermediary between the client and the <i>IRivRecordListener</i> specified by the user.
8. <i>CRivClientInterface.RCITimerCallback</i>	This is an inner class implementing the <i>RendezvousRvTimerCallback</i> interface. It abstracts out the Rendezvous layer, acting as an intermediary between the client and the user <i>IRivTimerCallback</i> object.
9. <i>CRivDbEntity</i>	This class is a representation of a database entity, with the general form "database.table.column".
10. <i>CRivDummyCallback</i>	This class is used as a dummy handler to process replies for certain queries.
11. <i>CRivEvalClause</i>	This class contains the information dictated in an eval clause to be found in active language such as AOC classes or STORE configs.
12. <i>CRivException</i>	This class contains exceptions which signal an error in a MWFM application.
13. <i>CRivException.RivErrRec</i>	This is the internal wrapper class for the error code, level and message.
14. <i>CRivExpr</i>	This class contains and evaluates expressions that are used as part of boolean query arithmetic.
15. <i>CRivFilter</i>	This class stores, in a tree format, queries such as ((y < 3) AND (x = 1)) OR (z != 0). Each node can be a left right subtree plus algebraic operator, or two query atoms (i.e the y < 3 bit) and algebraic combinations.
16. <i>CRivFilterParser</i>	This class defines an ultimate wrapper for the Filter parser, <i>RivFilter</i> , allowing a text <i>String</i> , in the appropriate format, to be parsed to a MWFM filter object, <i>CRivFilter</i> .
17. <i>CRivHashVector</i>	This class is an extension of <i>java.util.Hashtable</i> , which allows values to be extracted in the order in which they were added.

Class	Description
18. <i>CRivMonitorFilterParser</i>	This class is an extension of <i>CRivFilterParser</i> which allows the name fields in filters to include the style of addressing "Name[.x]+" i.e. a name, followed by one or more ".x", where x is any number, e.g. "ifOperStatus.1".
19. <i>CRivQueryAtom</i>	This class encodes the basic query on the data store.
20. <i>CRivRecord</i>	This class is the fundamental stored item in <i>all</i> MWFM data engines.
21. <i>CRivROMPer</i>	This class contains low level encode/decode methods for ROMP (RiverSoft Object Multicast Protocol) format packets.
22. <i>CRivRvDataHandler</i>	This handler class is used to receive and process data for a Rendezvous listener.
23. <i>CRivTransport</i>	A transport layer object used by every sender / receiver to process data to network, or network to data.
24. <i>CRivVarBind</i>	This class is used to store paired values, such as name=value.
25. <i>CRivVarBindList</i>	This class provides essentially the same functionality as <code>java.util.Vector</code> , but can only contain <i>CRivVarBinds</i> .
26. <i>IRivAlgebraic</i>	This interface defines the constants representing the different boolean algebraic operators.
27. <i>IRivConstants</i>	This interface contains a set of constants for attributes such as the standard field names for event records in riv_f_amos , model records in riv_model , or user and profile records in riv_auth .
28. <i>IRivDataType</i>	This interface defines the constants representing the different data types supported by <i>CRivAtom</i> .
29. <i>IRivNodeType</i>	This interface defines the constants representing the different types of nodes in a tree.
30. <i>IRivOper</i>	This interface defines the constants representing the different operations that can be performed on a tree.
31. <i>IRivRecordListener</i>	This interface defines the methods that an object must implement to "listen" for the arrival of records from the transport layer.
32. <i>IRivSubjects</i>	This interface defines the basic subject definitions used by all engines and clients in MWFM applications.
33. <i>IRivTimerCallback</i>	<i>IRivTimerCallback</i> defines the interface for handler classes of timer event activities in the client.
34. <i>ParseException</i>	This exception is thrown when parse errors are encountered.
35. <i>RivFilter</i>	This class defines the parser for filter objects, allowing a text <code>String</code> in the appropriate format to be parsed to a MWFM filter object, <i>CRivFilter</i> .

Class	Description
36. <i>RivFilterConstants</i>	This interface defines the tokens and keywords used by the Filter parser, <i>RivFilter</i> .
37. <i>RivFilterTokenManager</i>	This class manages the tokens and keywords used by the Filter parser, <i>RivFilter</i> .
38. <i>RivMonitorFilter</i>	This class defines the parser for filter objects, allowing a text <code>String</code> in the appropriate format to be parsed to a MWFM filter object, <i>CRivFilter</i> .
39. <i>RivMonitorFilterConstants</i>	This interface defines the tokens and keywords used by the Filter parser, <i>RivMonitorFilter</i> .
40. <i>RivMonitorFilterTokenManager</i>	This class manages the tokens and keywords used by the Filter parser, <i>RivMonitorFilter</i> .
41. <i>Token</i>	This class describes the input token stream. It is used by parsers, such as <i>RivFilter</i> . Application-level code should never need to use this class.
42. <i>TokenMgrError</i>	This class is used by parsers, such as <i>RivFilter</i> . Application-level code should never need to use this class.

Class/Interface reference pages

A key to the location of reference information for all classes and interfaces is presented in the table below:

Class	Reference Page
1. <i>ASCII_CharStream</i>	8
2. <i>CRivASNIAddress</i>	10
3. <i>CRivAtom</i>	16
4. <i>CRivClient</i>	26
5. <i>CRivClientHelper</i>	37
6. <i>CRivClientInterface</i>	55
7. <i>CRivClientInterface.RCIRecordListener</i>	61
8. <i>CRivClientInterface.RCITimerCallback</i>	63
9. <i>CRivDbEntity</i>	65
10. <i>CRivDummyCallback</i>	67
11. <i>CRivEvalClause</i>	69
12. <i>CRivException</i>	72
13. <i>CRivException.RivErrRec</i>	82
14. <i>CRivExpr</i>	84
15. <i>CRivFilter</i>	90
16. <i>CRivFilterParser</i>	96

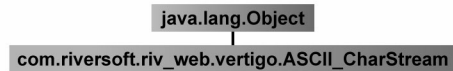
Class	Reference Page
17. <i>CRivHashVector</i>	100
18. <i>CRivMonitorFilterParser</i>	107
19. <i>CRivQueryAtom</i>	109
20. <i>CRivRecord</i>	114
21. <i>CRivROMPer</i>	122
22. <i>CRivRvDataHandler</i>	135
23. <i>CRivTransport</i>	141
24. <i>CRivVarBind</i>	150
25. <i>CRivVarBindList</i>	154
26. <i>IRivAlgebraic</i>	175
27. <i>IRivConstants</i>	176
28. <i>IRivDataType</i>	200
29. <i>IRivNodeType</i>	203
30. <i>IRivOper</i>	204
31. <i>IRivRecordListener</i>	209
32. <i>IRivSubjects</i>	210
33. <i>IRivTimerCallback</i>	217
34. <i>ParseException</i>	218
35. <i>RivFilter</i>	222
36. <i>RivFilterConstants</i>	225
37. <i>RivFilterTokenManager</i>	227
38. <i>RivMonitorFilter</i>	228
39. <i>RivMonitorFilterConstants</i>	232
40. <i>RivMonitorFilterTokenManager</i>	234
41. <i>Token</i>	235
42. <i>TokenMgrError</i>	239

Class ASCII_CharStream

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-2 Hierarchy of class ASCII_CharStream



```

public final class ASCII_CharStream
extends java.lang.Object
  
```

Description

An input stream class used by parsers, where the stream is assumed to contain only ASCII characters (without unicode processing).

See Also

RivFilter



Note

ASCII_CharStream This class is only used by the parsers—CRivFilterParser and CRivMonitorFilterParser. Application code should never need to use this class directly. The summaries are included for reference only and thus no descriptions are included

Field summary

Field

1. public int *bufpos*
 2. public static final boolean *staticFlag*
-

Constructor summary

Constructor
1. <code>public ASCII_CharStream(java.io.InputStream dstream, int startline, int startcolumn)</code>
2. <code>public ASCII_CharStream(java.io.InputStream dstream, int startline, int startcolumn, int buffersize)</code>
3. <code>public ASCII_CharStream(java.io.Reader dstream, int startline, int startcolumn)</code>
4. <code>public ASCII_CharStream(java.io.Reader dstream, int startline, int startcolumn, int buffersize)</code>

Method summary

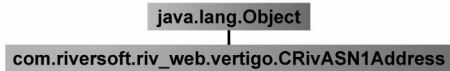
Method
1. <code>public void adjustBeginLineColumn(int newLine, int newCol)</code>
2. <code>public final void backup(int amount)</code>
3. <code>public final char BeginToken()</code>
4. <code>public void Done()</code>
5. <code>public final int getBeginColumn()</code>
6. <code>public final int getBeginLine()</code>
7. <code>public final int getColumn()</code>
8. <code>public final int getEndColumn()</code>
9. <code>public final int getEndLine()</code>
10. <code>public final String GetImage()</code>
11. <code>public final int getLine()</code>
12. <code>public final char[] GetSuffix(int len)</code>
13. <code>public final char readChar()</code>
14. <code>public void ReInit(java.io.InputStream dstream, int startline, int startcolumn)</code>
15. <code>public void ReInit(java.io.InputStream dstream, int startline, int startcolumn, int buffersize)</code>
16. <code>public void ReInit(java.io.Reader dstream, int startline, int startcolumn)</code>
17. <code>public void ReInit(java.io.Reader dstream, int startline, int startcolumn, int buffersize)</code>

Class CRivASN1Address

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-3 Hierarchy of class CRivASN1Address



```
public class CRivASN1Address
extends java.lang.Object
```

Description

The class `CRivASN1Address` contains hierarchical style addressing, such as 1.2.3.4, and is used to hang both the topology model and CLASS model inside of all engines that need it (particularly the `riv_disco` engine).

Field summary

Field	Reference Page
1. public static int <i>RIV_MAX_ADDR_DEPTH</i>	12

Constructor summary

Constructor	Reference Page
1. public <i>CRivASN1Address</i> ()	12
2. public <i>CRivASN1Address</i> (<i>CRivASN1Address</i> arg)	12
3. public <i>CRivASN1Address</i> (<i>CRivASN1Address</i> s parent, int child)	13
4. public <i>CRivASN1Address</i> (int firstNEntries, <i>CRivASN1Address</i> arg)	13
5. public <i>CRivASN1Address</i> (int relative, int[] full, int depth)	14
6. public <i>CRivASN1Address</i> (String printable)	14

Method summary

Method	Reference Page
1. public int <i>getRAAAddressAt</i> (int position)	14
2. public int <i>getRAADepth</i> ()	14
3. public int[] <i>getRAAFullAddress</i> ()	15
4. public int <i>getRAARelativeAddress</i> ()	15
5. public boolean <i>isRAAMatchAddress</i> (<i>CRivASN1Addr</i> ess arg)	15
6. public boolean <i>isRAAPartialMatch</i> (<i>CRivASN1Addr</i> ess arg)	15

Method	Reference Page
7. <code>public CRivASN1Address <i>raaUnion</i>(CRivASN1Address base, CRivASN1Address qual)</code>	16
8. <code>public String <i>toString</i>()</code>	16

Field detail

1)

Field	<code>public static int RIV_MAX_ADDR_DEPTH</code>
Description	The maximum length of a CRivASN1Address.
Value	255

Constructor detail

1)

Constructor	<code>public CRivASN1Address()</code>
Description	Default constructor.

2)

Constructor	<code>public CRivASN1Address(CRivASN1Address arg)</code>
Description	Copy constructor.
Parameters	<code>arg</code> —the CRivASN1Address to copy.

3)

Constructor	<code>public CRivASN1Address(CRivASN1Address parent, int child)</code>
Description	Construct a <code>CRivASN1Address</code> from another <code>CRivASN1Address</code> and a child. The new address will have the child integer tagged onto the end of the supplied <code>CRivASN1Address</code> parameter. For example, supplying an address in the form 1.2.3.4 and a child <code>int</code> of 5 would construct a new address in the form 1.2.3.4.5. Be careful not to confuse with the parameterized copy constructor.
Parameters	<code>parent</code> —the initial value that will have the child integer tagged onto it. <code>child</code> —the extra value being added.
See Also	<code>CRivASN1Address(int, CRivASN1Address)</code> , <code>toString</code>

4)

Constructor	<code>public CRivASN1Address(int firstNEntries, CRivASN1Address arg)</code>
Description	Create a new <code>CRivASN1Address</code> by copying the first “n” entries of the supplied <code>CRivASN1Address</code> . For example, supplying an address in the form 1.2.3.4.5 and <code>firstNEntries</code> of 4 would construct a new address in the form 1.2.3.4. Be careful not to confuse with the constructor taking an address and child.
Parameters	<code>firstNEntries</code> —the number of entries that will be copied from the supplied <code>CRivASN1Address</code> . <code>arg</code> —the <code>CRivASN1Address</code> whose first “n” entries will be copied.
Throws	<code>ArrayIndexOutOfBoundsException</code> —if <code>firstNEntries</code> is greater than the address depth of the supplied address.
See Also	<code>CRivASN1Address(int, CRivASN1Address)</code> , <code>getRAADepth()</code>

5)

Constructor	<code>public CRivASN1Address(int relative, int full, int depth)</code>
Description	Make a <code>CRivASN1Address</code> , from a relative address stub and an array of integers, which will make up the new address.
Parameters	<code>relative</code> —the relative address stub. <code>full</code> —the array of <code>ints</code> which will make up the <code>CRivASN1Address</code> . <code>depth</code> —the depth of the new address.

6)

Constructor	<code>public CRivASN1Address(String printable)</code>
Description	Construct a new <code>CRivASN1Address</code> by parsing the given <code>String</code> , which must be in the format <code>u.v.w.x.y.z</code> (which has a maximum number of values equal to <code>RIV_MAX_ADDR_DEPTH</code>) such as “1.2.3.4”.
Parameters	<code>printable</code> —the <code>String</code> representation of the new address.
See Also	<code>RIV_MAX_ADDR_DEPTH</code>

Method detail

1)

Method	<code>public int getRAAAddressAt(int position)</code>
Description	Get the relative name at the specified position. For example, in address 1.2.3.4, <code>getRAAAddressAt(2)</code> returns 3.
Parameter	<code>position</code> —the position at which to get the address.
Returns	-1 if there is no address at the specified position.

2)

Method	<code>public int getRAADepth()</code>
Description	Get the address depth.
Returns	The address depth.

3)

Method `public int getRAAFullAddress()`

Description Return the full address as an array of `ints`.

4)

Method `public int getRAARelativeAddress()`

Description Return the relative address stub of this address, which will be zero, unless set in one of the constructors which take a relative address parameter, or combine a parent address and a child.

5)

Method `public boolean isRAAMatchAddress(CRivASN1Address arg)`

Description Compare two `CRivASN1Addresses` to see if they represent the same address. This is done by comparing the `String` representations of the addresses.

Parameter `arg`—the address to match with ‘`this`’ address.

Returns `true`, if the two `CRivASN1Addresses` match, `false` otherwise.

6)

Method `public boolean isRAAPartialMatch(CRivASN1Address arg)`

Description Compare this `CRivASN1Address` to see if it matches the first “`n`” entries of a supplied address, where “`n`” is equal to the address depth of this address.

Parameter `arg`—the address to match with ‘`this`’ address.

Returns `true`, if the first `n` entries of the supplied address match this address. The method will return `false` if the address depth of the supplied address is less than the depth of this address.

7)

Method `public CRivASN1Address raaUnion(CRivASN1Address base, CRivASN1Address qual)`

Description Make a `CRivASN1Address` by combining two `CRivASN1Addresses`.

Parameters `base`—the base address to combine.
`qual`—the qualified address to combine.

Returns The address made by combining the two addresses.

8)

Method `public String toString()`

Description Return a `String` representation of this `CRivASN1Address` in a `String` format e.g. 1.0.0.0.6.8.0

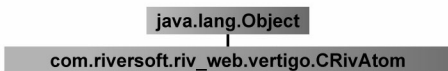
Overrides `toString` in class `java.lang.Object`.

Class CRivAtom

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Example 4-1 Hierarchy of class CRivAtom



Note

`CRivAtom`—In order to increase the speed of execution, most of the methods in this class that return data from within the atom do not check the type of data. Instead, if an incorrect method is used, such as calling `getRAASN1Address()` when the data contained is actually an int, the method will throw a `ClassCastException`. Thus, if there is any doubt as to the type of data contained within the atom, always query the type (using `getRAType()`) before retrieving a value.

```

public class CRivAtom
extends java.lang.Object
implements IRivDataType, IRivOper
  
```

Description

This is the primary container class throughout MWFM applications. It is essentially a wrapper around any of the fundamental data types defined in IRivDataType such as String, int, or CRivASN1Address. It can also act as a container for a Vector of other CRivAtom objects, or a list of CRivVarBinds, held in a CRivVarBindList. The former of these two list types is often referred to as a "LIST", while the latter is referred to as an "OBJECT".

This class also provides methods for comparisons between two CRivAtoms using the operators defined in the interface IRivOper.

See Also

CRivASN1Address, CRivVarBindList, IRivDataType, IRivOper.

Constructor summary

Constructor	Reference Page
1. public <i>CRivAtom</i> ()	19
2. public <i>CRivAtom</i> (byte[] value)	19
3. public <i>CRivAtom</i> (CRivASN1Address value)	19
4. public <i>CRivAtom</i> (CRivAtom atomToCopy)	19
5. public <i>CRivAtom</i> (CRivVarBindList value)	20
6. public <i>CRivAtom</i> (float value)	20
7. public <i>CRivAtom</i> (int value)	20
8. public <i>CRivAtom</i> (long value)	20
9. public <i>CRivAtom</i> (String value)	20
10. public <i>CRivAtom</i> (java.util.Vector value)	21

Method summary

Method	Reference Page
1. public boolean <i>equals</i> (Object obj)	21
2. public CRivASN1Address <i>getRAASN1Address</i> ()	21
3. public byte[] <i>getRAByteArray</i> ()	21
4. public java.lang.Class <i>getRADataClass</i> ()	22
5. public float <i>getRAFloatValue</i> ()	22
6. public int <i>getRAIntValue</i> ()	22
7. public long <i>getRALongValue</i> ()	22
8. public CRivVarBindList <i>getRAObject</i> ()	22
9. public CRivAtom <i>getRASubVal</i> (String name)	23
10. public int <i>getRAType</i> ()	23
11. public java.util.Vector <i>getRAVector</i> ()	23
12. public boolean <i>isRAAtomInVector</i> (CRivAtom entry)	23
13. public CRivAtom <i>raDeepCopyListAtom</i> ()	24
14. public CRivAtom <i>raDeepCopyObjectAtom</i> ()	24
15. public boolean <i>raEvalRelop</i> (int operator, CRivAtom atomToCompare)	24
16. public int <i>raLex</i> (CRivAtom comp)	25
17. public void <i>setRAValue</i> (CRivASN1Address value)	25
18. public void <i>setRAValue</i> (CRivVarBindList value)	25
19. public void <i>setRAValue</i> (float value)	25
20. public void <i>setRAValue</i> (int value)	25
21. public void <i>setRAValue</i> (String value)	26
22. public void <i>setRAValue</i> (java.util.Vector value)	26
23. public String <i>toString</i> ()	26

Constructor detail

1)

Constructor `public CRivAtom()`

Description Default constructor.

2)

Constructor `public CRivAtom(byte[] value)`

Description Make one based on an array of bytes.

Parameters `value`—the initial value for this `CRivAtom`.

3)

Constructor `public CRivAtom(CRivASN1Address value)`

Description Make one based on a `CRivASN1Address`.

Parameters `value`—the initial value for this `CRivAtom`.

See Also `CRivASN1Address`

4)

Constructor `public CRivAtom(CRivAtom atomToCopy)`

Description Creates a deep copy of the values held in the argument `CRivAtom`.

Parameters `atomToCopy`—the atom to be deep copied.

Throws `java.lang.ClassCastException`—if the `atomToCopy` is of type `IRivDataType.RDT_LIST` or `IRivDataType.RDT_OBJECT`, and this is not a vector of `CRivAtoms`, or `CRivVarBinds`.

5)

Constructor	<code>public CRivAtom(CRivVarBindList value)</code>
Description	Make one based on a <code>CRivVarBindList</code> . The <code>CRivAtom</code> will be of type <code>IRivDataType.RDT_OBJECT</code> .
Parameters	<code>value</code> —the initial value of the <code>CRivAtom</code> .

6)

Constructor	<code>public CRivAtom(float value)</code>
Description	Make one based on a <code>float</code> .
Parameters	<code>value</code> —the initial value for this <code>CRivAtom</code> .

7)

Constructor	<code>public CRivAtom(int value)</code>
Description	Make one based on an <code>int</code> .
Parameters	<code>value</code> —the initial value for this <code>CRivAtom</code> .

8)

Constructor	<code>public CRivAtom(long value)</code>
Description	Make one based on a <code>long</code> .
Parameters	<code>value</code> —the initial value for this <code>CRivAtom</code> .

9)

Constructor	<code>public CRivAtom(String value)</code>
Description	Make one based on a <code>String</code> .
Parameters	<code>value</code> —the initial value for this <code>CRivAtom</code> .

10)

Constructor	<code>public CRivAtom(java.util.Vector value)</code>
Description	Make one based on a <code>Vector</code> . The <code>CRivAtom</code> will be of type <code>IRivDataType.RDT_LIST</code> .
Parameters	<code>value</code> —the initial value for this <code>CRivAtom</code> .
Throws	<code>java.lang.ClassCastException</code> —if the <code>Vector</code> is not a list of <code>CRivAtom</code> objects.

Method detail**1)**

Method	<code>public boolean equals(Object obj)</code>
Description	Compares this <code>CRivAtom</code> to the specified object. Overrides <code>equals(Object obj)</code> in class <code>Object</code> . The result is <code>true</code> if, and only if, the argument is not <code>null</code> and is a <code>CRivAtom</code> object that represents the same <code>CRivAtom</code> as this object. Uses the <code>raLex()</code> method to test for lexicographical equality.
Parameters	<code>obj</code> —the object to compare with.
Overrides	<code>equals</code> in class <code>java.lang.Object</code> .

2)

Method	<code>public CRivASN1Address getRAASN1Address()</code>
Description	Return the value of this <code>CRivASN1Address</code> atom.
Throws	<code>java.lang.ClassCastException</code> —if the <code>CRivAtom</code> is not of type <code>IRivDataType.RDT_ASN1</code> .
See Also	<code>CRivASN1Address</code>

3)

Method	<code>public byte[] getRAByteArray()</code>
Description	Return the value of this atom, which is an array of bytes.
Throws	<code>java.lang.ClassCastException</code> —if the <code>CRivAtom</code> is not of type <code>IRivDataType.RDT_BYTE_ARRAY</code> .

4)

Method `public java.lang.Class getRADataClass()`

Description Return the runtime class of the value of this CRivAtom.

Throws `java.lang.NullPointerException`—if the CRivAtom has null data.

5)

Method `public float getRAFloatValue()`

Description Return the value of this float atom.

Throws `java.lang.ClassCastException`—if the CRivAtom is not of type `IRivDataType.RDT_FLOAT`.

6)

Method `public int getRAIntValue()`

Description Return the value of this int atom.

Throws `java.lang.ClassCastException`—if the CRivAtom is not of type `IRivDataType.RDT_INTEGER`.

7)

Method `public long getRALongValue()`

Description Return the value of this long atom.

Throws `java.lang.ClassCastException`—if the CRivAtom is not of type `IRivDataType.RDT_LONG`.

8)

Method `public CRivVarBindList getRAObject()`

Description Return the value of this Object atom, which is a `CRivVarBindList`.

Throws `java.lang.ClassCastException`—if the CRivAtom is not of type `IRivDataType.RDT_OBJECT`.

9)

Method `public CRivAtom getRASubVal(String name)`

Description Convenient method to extract a value from a given name from a `IRivDataType.RDT_OBJECT CRivAtom`. Returns null if the `CRivAtom` is not of type `IRivDataType.RDT_OBJECT`, or if the `IRivDataType.RDT_OBJECT` does not contain the named field.

Parameters `name`—the name of the `CRivVarBind` to extract the value from.

10)

Method `public int getRAType()`

Description Return the data type of this atom. Uses the constants defined in `IRivDataType` to specify different atom types.

11)

Method `public java.util.Vector getRAVector()`

Description Return the value of this Vector atom, which is a `java.util.Vector` of `CRivAtom` objects.

Throws `java.lang.ClassCastException`—if the `CRivAtom` is not of type `IRivDataType.RDT_LIST`.

12)

Method `public boolean isRAAtomInVector(CRivAtom entry)`

Description Is the supplied atom (`entry`) in 'this' atom? Where `this` must be of type `IRivDataType.RDT_LIST` and `entry` can be any type.

Parameters `entry`—the atom to check if it is in `this` atom's Vector.

Returns `true`, if the supplied `CRivAtom` is in `this` atom, `false` if the supplied `CRivAtom` is not in `this` atom, or if `this` atom is not of type `IRivDataType.RDT_LIST`.

13)

Method `public CRivAtom raDeepCopyListAtom()`

Description Make a deep copy of a `CRivAtom` which is of type `IRivDataType.RDT_LIST`.

Throws `java.lang.ClassCastException`—if this `CRivAtom` is not of type `IRivDataType.RDT_LIST`.

Returns The deep copy of `this` atom.

14)

Method `public CRivAtom raDeepCopyObjectAtom()`

Description Make a deep copy of a `CRivAtom` which is of type `IRivDataType.RDT_OBJECT`.

Throws `java.lang.ClassCastException`—if this `CRivAtom` is not of type `IRivDataType.RDT_OBJECT`.

Returns The deep copy of `this` atom.

15)

Method `public boolean raEvalRelop(int operator, CRivAtom atomToCompare)`

Description Compare `this` atom with the supplied atom parameter.

Parameters `atomToCompare`—the `CRivAtom` to compare with `this` atom.
`operator`—the type of operator to use in the comparison, must be one of the values defined in `IRivOper`.

Returns `true`, if `this` atom and `atomToCompare` satisfy the condition specified by the operator.

16)

Method `public int raLex(CRivAtom comp)`

Description Performs a lexical comparison of `this` atom with the supplied atom parameter.

Parameters `comp`—the `CRivAtom` to compare with.

Returns 0 if they represent the same data lexicographically, +/- otherwise.

17)

Method `public void setRAValue(CRivASN1Address value)`

Description Set a new value to this atom.

Parameters `value`—the new value to be set.

18)

Method `public void setRAValue(CRivVarBindList value)`

Description Set a new value to this atom.

Parameters `value`—the new value to be set.

19)

Method `public void setRAValue(float value)`

Description Set a new value to this atom.

Parameters `value`—the new value to be set.

20)

Method `public void setRAValue(int value)`

Description Set a new value to this atom.

Parameters `value`—the new value to be set.

21)

Method `public void setRAValue(String value)`

Description Set a new value to this atom.

Parameters `value`—the new value to be set.

22)

Method `public void setRAValue(java.util.Vector value)`

Description Set a new value to this atom.

Parameters `value`—the new value to be set.

Throws `java.lang.ClassCastException`—if the `Vector` is not a list of `CRivAtom` objects.

23)

Method `public String toString()`

Description Return a `String` representation of this atom.

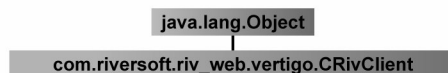
Overrides `toString` in class `java.lang.Object`

Class CRivClient

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-4 Hierarchy of class *CRivClient*



```

public class CRivClient
extends java.lang.Object
  
```


Description

This basic class manages a session to Rendezvous, handles initialization and client handling. A client is created to handle requests from a given MWFM domain. Rendezvous sessions can be started for Java applications or applets, using the relevant `rcInitSession()` method. Requests are then made on certain MWFM subjects, as defined by constants in the interface `IRivSubjects`. For example, to make queries for information from the topology store, `riv_model`, or to request updates from the event store `riv_f_amos`.

Many common requests and operations can be handled through the helper class `CRivClientInterface`, so that the user does not have to call any of the methods of this class directly after initialization.

See Also

`IRivSubjects`

Field summary

Field	Reference Page
1. <code>public static final long RIV_DEFAULT_LATENCY</code>	29

Constructor summary

Constructor	Reference Page
1. <code>public CRivClient(long timeOut, String domain)</code>	29

Method summary

Method	Reference Page
1. public String <i>getRCDomain</i> ()	29
2. public String <i>getRCDomainSubj</i> (String subjStub)	30
3. public long <i>getRCTimeOut</i> ()	30
4. public boolean <i>isRCValidOutboundType</i> (Object data)	30
5. public COM.TIBCO.rv.RvTimer <i>rcAddOnceOnlyTimer</i> (long interval, COM.TIBCO.rv.RvTimerCallback handler)	31
6. public COM.TIBCO.rv.RvTimer <i>rcAddRepeatTimer</i> (long interval, COM.TIBCO.rv.RvTimerCallback handler)	31
7. public int <i>rcAddService</i> (String subject, COM.TIBCO.rv.RvDataCallback updateHandler)	32
8. public int <i>rcInitSession</i> ()	32
9. public int <i>rcInitSession</i> (String hostname)	32
10. public int <i>rcInitSession</i> (String hostname, int port)	33
11. public int <i>rcInitSession</i> (String service, String network, String daemon)	33
12. public void <i>rcRemoveTimer</i> (RvTimer timer)	34
13. public int <i>rcSend</i> (String subject, Object data)	35
14. public int <i>rcSend</i> (String subject, Object data, COM.TIBCO.rv.RvDataCallback handler)	35
15. public int <i>rcTerminateSession</i> ()	36
16. public int <i>rcTimedSend</i> (String subject, Object data, long timeout, COM.TIBCO.rv.RvDataCallback handler)	36
17. public void <i>setRCTimeOut</i> (long timeout)	36

Field detail

1)

Field	<code>public static final long RIV_DEFAULT_LATENCY</code>
Description	Define a default value for the latency, in milliseconds.
Value	3000

Constructor detail

1)

Constructor	<code>public CRivClient(long timeOut, String domain)</code>
Description	Create a new client object to manage a session to Rendezvous, with the specified timeout period (in milliseconds) for a response, and for the specified domain.
Parameters	<code>timeOut</code> —the timeout latency held by this <code>CRivClient</code> , in milliseconds. <code>domain</code> —a named view consisting of a single set of core NMOS applications.

Method detail

1)

Method	<code>public String getRCDomain()</code>
Description	Return a reference to the domain name, where a domain is a named “view” of a single set of core NMOS components. Each domain has its own set of users and passwords and manages a particular configured view of a network.

2)

Method `public String getRCDomainSubj(String subjStub)`

Description Construct a `String` for sending to the Rendezvous host, incorporating the supplied MWFm subject stub and this client's domain.

Parameters `subjStub`—the subject stub.

Returns A `String` which incorporates the subject stub and the client's domain, where the subject stub should be one of the values defined in `IRivSubjects`.

3)

Method `public long getRCTimeOut()`

Description Returns the default latency, or time out interval, in milliseconds, of the client object. This value can be used in methods where a timed listener will be created for replies. A timed listener expects to receive a data message before its time limit expires. If no message is received within that time then the TIB/Rendezvous event manager will call the `onTimeOut ()` method of the listeners handler object.

Returns The time out, in milliseconds.

4)

Method `public boolean isRCValidOutboundType(Object data)`

Description Determine whether the specified data object is a valid outbound data type. Only certain classes can be encoded into valid TIB/Rendezvous messages for outbound transport, which are `String`, `Boolean`, `Byte`, `Short`, `Integer`, `Long`, `Float`, `Double`, `Date`, `java.net.InetAddress` or `byte []`.

Parameters `data`—the data to be validated as an outbound type.

Returns `true`, if the class of the specified data is a valid outbound data type, `false` otherwise.

5)

Method	<code>public COM.TIBCO.rv.RvTimer rcAddOnceOnlyTimer(long interval, COM.TIBCO.rv.RvTimerCallback handler)</code>
Description	Add a once-only timer. Rendezvous has a timer despatch thread which triggers the <code>onTimer()</code> <code>RvTimerCallback</code> method. Note The user should not need to call this method directly. Instead, the equivalent method in <code>CRivClientInterface</code> should be used (<code>rciAddOnceOnlyTimer</code>).
Parameters	<code>interval</code> —the timer trigger interval, in milliseconds, which must be > 0. <code>handler</code> —the object to handle the timer event activity, which must be non-null.
Returns	The <code>RvTimer</code> that was created, null if a <code>RvTimer</code> was not successfully created.
Throws	<code>CRivException</code> —if the TIB/Rendezvous session is not valid, or the timer could not be added.

6)

Method	<code>public COM.TIBCO.rv.RvTimer rcAddRepeatTimer(long interval, COM.TIBCO.rv.RvTimerCallback handler)</code>
Description	Add a repeating timer. Rendezvous has a timer despatch thread which triggers the <code>onTimer()</code> <code>RvTimerCallback</code> method. Note The user should not need to call this method directly. Instead, the equivalent method in <code>CRivClientInterface</code> should be used (<code>rciAddRepeatTimer</code>).
Parameters	<code>interval</code> —the timer trigger interval, in milliseconds, which must be > 0. <code>handler</code> —the object to handle the timer event activity which must be non-null.
Returns	The <code>RvTimer</code> that was created; null if a <code>RvTimer</code> was not successfully created.
Throws	<code>CRivException</code> —if the TIB/Rendezvous session is not valid, or the timer could not be added.

7)

Method	<code>public int rcAddService(String subject, COM.TIBCO.rv.RvDataCallback updateHandler)</code>
Description	Add a listener to listen for updates from Rendezvous on the specified subject.
Parameters	<code>subject</code> —one of the values from <code>IRivSubjects</code> . <code>updateHandler</code> —the object implementing the <code>RvDataCallback</code> interface to process any incoming messages.
Returns	An <code>int</code> , one of the constants defined in <code>CRivException</code> , signalling whether the listener was added successfully, e.g., <code>CRivException.RIV_OK</code> if successful.
Throws	<code>CRivException</code> —if the TIB/Rendezvous session is not valid.

8)

Method	<code>public int rcInitSession()</code>
Description	Initialize a Rendezvous <code>rvd</code> session for an independent Java application.
Returns	An <code>int</code> , one of the constants defined in <code>CRivException</code> , signalling whether the <code>RvSession</code> was created successfully, e.g., <code>CRivException.RIV_OK</code> if successful.

9)

Method	<code>public int rcInitSession(String hostname)</code>
Description	Attempts to connect to a Rendezvous <code>rva</code> session for Java applets. TCP port 7600 is used as the default port. The applet cannot start an <code>rva</code> process, but must connect to an agent process that is already running. This can be started at the command line by typing the command “ <code>rva - flavor 116</code> ”, after having previously killed off any existing <code>rvd</code> sessions.
Parameters	<code>hostname</code> —the Rendezvous agent (<code>rva</code>) to connect to that is running on the computer with this hostname.
Returns	An <code>int</code> , one of the constants defined in <code>CRivException</code> , signalling whether the <code>RvSession</code> was created successfully, e.g., <code>CRivException.RIV_OK</code> if successful.

10)

- Method** `public int rcInitSession(String hostname, int port)`
- Description** Attempts to connect to a Rendezvous `rva` session for Java applets. The applet cannot start a `rva` process, but must connect to an agent process that is already running. This can be started at the command line using the command “`rva - flavor 116`”, after having previously killed off any existing `rvd` sessions.
- Parameters** `hostname`—the Rendezvous agent (`rva`) to connect to that is running on the computer with this hostname.
`port`—the Rendezvous agent (`rva`) on this TCP port to connect to. This value must match the `-listen` parameter of `rva` (the `-listen` parameter of the Rendezvous daemon specifies where the Rendezvous daemon should listen for new application sessions).
- Returns** An `int`, one of the constants defined in `CRivException`, signalling whether the `RvSession` was created successfully, e.g., `CRivException.RIV_OK` if successful.

11)

- Method** `public int rcInitSession(String service, String network, String daemon)`
- Description** Initialize a `rvd` session, with the specified service, network and daemon parameters.

Parameters `service`—specifies the service group to communicate with; `null` specifies the default Rendezvous service. The Rendezvous daemon divides the network into service groups. Each session belongs to a single service group and a session can only communicate with other sessions in the same service group. To communicate with more than one service group, applications must initialize more than one session.

`network`—instructs the Rendezvous daemon to use a particular network for all communications involving this session; `null` specifies the primary network interface for the host computer. Every application session communicates with other sessions over a single network. On computers with more than one network interface, this parameter instructs the Rendezvous daemon to use a particular network for all communications involving this session. To communicate over more than one network, applications must initialize more than one session.

`daemon`—instructs a `rvd` session about how and where to find the Rendezvous daemon and establish communication; `null` specifies the default - find the local daemon on TCP port 45001.

Returns An `int`, one of the constants defined in `CRivException`, signalling whether the `RvSession` was created successfully, e.g., `CRivException.RIV_OK` if successful.

12)

Method `public void rcRemoveTimer(RvTimer timer)`

Description Remove timer activity to cancel interest in the corresponding event.

A once-only timer can be removed before it triggers its handler's callback method, and it should be noted that triggering will automatically remove it, so that calling this method would be unnecessary.

Removing any timer that has already been removed has no effect.

Note The user should not need to call this method directly. Instead, the equivalent method in `CRivClientInterface` should be used (`rciRemoveTimer`).

Parameters `timer`—the timer to be removed.

13)

Method	<code>public int rcSend(String subject, Object data)</code>
Description	Send a message to a Rendezvous domain, using a specific subject.
Parameters	<code>subject</code> —the destination subject for which to send the message. This should be one of the values from <code>IRivSubjects</code> . <code>data</code> —the object to send as the contents of the message.
Returns	An <code>int</code> , one of the constants defined in <code>CRivException</code> , signalling whether the data was sent successfully, e.g., <code>CRivException.RIV_OK</code> if successful.

14)

Method	<code>public int rcSend(String subject, Object data, COM.TIBCO.rv.RvDataCallback handler)</code>
Description	Send a message to a Rendezvous domain, using a specific subject, specifying the listener to handle the reply. No time limit is set for the listener to receive the reply. The specified handler object to listen for replies should be non-null or the method will return <code>CRivException.NULL_POINTER_REF</code> .
Parameters	<code>subject</code> —the destination subject for which to send the message. This should be one of the values from <code>IRivSubjects</code> . <code>data</code> —the object to send as the contents of the message. <code>handler</code> —the object implementing the Rendezvous <code>RvDataCallback</code> interface, which will process any replies.
Returns	An <code>int</code> , one of the constants defined in <code>CRivException</code> , signalling whether the data was sent successfully, e.g., <code>CRivException.RIV_OK</code> if successful.
See Also	<code>rcTimedSend(String, Object, long, COM.TIBCO.rv.RvDataCallback)</code>

15)

Method `public int rcTerminateSession()`

Description Terminate the Rendezvous session, severing its connection to the TIB/Rendezvous daemon (*rvd*) or agent (*rva*). It is recommended that all programs call `rcTerminateSession()` before deleting the last reference to a session object.

Returns An `int`, one of the constants defined in `CRivException`, signalling whether the session was terminated successfully, e.g., `CRivException.RIV_OK` if successful.

16)

Method `public int rcTimedSend(String subject, Object data, long timeout, COM.TIBCO.rv.RvDataCallback handler)`

Description Send a message to a Rendezvous domain, using a specific subject, specifying the listener to handle the reply. The listener that handles the request will use the specified time limit for reply. If the time limit expires before the listener receives a message, the event manager will call the `onTimeout()` method of the handler object.

The specified handler should be non-null or the method will return `CRivException.NULL_POINTER_REF`.

Parameters `subject`—one of the values from `IRivSubjects`.
`data`—the object to send.
`timeout`—the time limit, in milliseconds, in which to receive a reply.
`handler`—the object implementing the `RvDataCallback` interface to process any incoming messages.

Returns An `int`, one of the constants defined in `CRivException`, signalling whether the data was sent successfully, e.g., `CRivException.RIV_OK` if successful.

17)

Method `public void setRCTimeOut(long timeout)`

Description Set the timeout limit held in this `CRivClient` that can be used in the methods that take a latency parameter.

Parameters `timeout`—the timeout latency held by this `CRivClient`, in milliseconds.

Class CRivClientHelper

(com.riversoft.riv_web.vertigo)

Hierarchy



```
public class CRivClientHelper
  extends CRivClientInterface
  implements IRivConstants
```

Description

This class is an extension of `CRivClientInterface` and acts as an intermediary between a `CRivClient` object and user defined classes. It provides convenience methods to allow polling of an active MWFM domain and abstracts out the Rendezvous methods and objects. It contains methods to obtain specific pieces of information from, or to insert, update or delete records from `riv_f_amos`, `riv_model` and `riv_auth`.

The constructor of this class requires the application level code to provide a valid user name and password for a user currently held in `riv_auth`. This is required in order to obtain certain pieces of information from `riv_auth`.

See Also

`CRivClient`

Constructor summary

Constructor	Reference Page
1. <code>public CRivClientHelper(CRivClient client, String username, String password)</code>	39

Method summary

Method	Reference Pages
1. <code>public CRivRecord[] getRCHAllEventRecords()</code>	40
2. <code>public CRivRecord[] getRCHAllModelRecords()</code>	40
3. <code>public boolean getRCHAuthPermission(String category, String action)</code>	41
4. <code>public CRivRecord[] getRCHAuthProfiles()</code>	41
5. <code>public CRivRecord[] getRCHAuthUsers()</code>	42
6. <code>public CRivRecord[] getRCHEventRecords(CRivFilter filter)</code>	42
7. <code>public CRivRecord[] getRCHEventRecords(CRivVarBindList vbList)</code>	43
8. <code>public CRivRecord[] getRCHInterfaces(CRivRecord modelRec)</code>	43
9. <code>public CRivRecord[] getRCHModelRecords(CRivFilter filter)</code>	44
10. <code>public CRivRecord[] getRCHModelRecords(CRivVarBindList vbList)</code>	44
11. <code>protected byte[] getRCHPacketForAuth(String qryString)</code>	45
12. <code>protected CRivRecord[] getRCHQueryRecords(String subject, Object qryData)</code>	45
13. <code>public void rchAddAmosListener(IRivRecordListener recListener)</code>	46
14. <code>public void rchAddModelListener(IRivRecordListener recListener)</code>	46
15. <code>public boolean rchClearEventRecord(CRivRecord eventRec)</code>	47
16. <code>public boolean rchDeleteModelRecord(CRivRecord modelRec)</code>	47
17. <code>public boolean rchDeleteProfileRecord(CRivRecord profileRec)</code>	48

Method	Reference Pages
18. <code>public boolean rchDeleteUserRecord (CRivRecord userRec)</code>	48
19. <code>public boolean rchInsertEventRecord (CRivRecord eventRec)</code>	49
20. <code>public boolean rchInsertModelRecord (CRivRecord modelRec)</code>	49
21. <code>public boolean rchInsertProfileRecord (CRivRecord profileRec)</code>	50
22. <code>public boolean rchInsertUserRecord (CRivRecord userRec)</code>	51
23. <code>public void rchRemoveAmosListener (IRivRecordListener recListener)</code>	51
24. <code>public void rchRemoveModelListener (IRivRecordListener recListener)</code>	52
25. <code>public boolean rchUpdateEventRecord (CRivRecord eventRec, CRivVarBindList evVbList)</code>	52
26. <code>public boolean rchUpdateModelRecord (CRivRecord modelRec, CRivVarBindList modVbList)</code>	53
27. <code>public boolean rchUpdateProfileRecord (CRivRecord profileRec, CRivVarBindList modVbList)</code>	53
28. <code>public boolean rchUpdateUserRecord (CRivRecord userRec, CRivVarBindList modVbList)</code>	54

Constructor detail

1)

Constructor	<code>public CRivClientHelper (CRivClient client, String username, String password)</code>
Description	Construct a <code>CRivClientHelper</code> for the specified <code>CRivClient</code> object. A username and password must also be provided as they define what information can be accessed from the client.
Throws	<code>java.lang.NullPointerException</code> —if the username or password are null.
See Also	<code>CRivClient</code>

Method detail

1)

Method `public CRivRecord[] getRCHAllEventRecords ()`

Description Return an array of all the event records currently held in the `mojo.events` table of **riv_f_amos**.

After sending the data, the client engine will expect to receive a response from **riv_f_amos** within the time limit specified by the latency timeout of the client, or the method will throw a `CRivException`.

This method is blocking so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.

Returns An array containing all the event records, as `CRivRecord` objects in the `mojo.events` table of **riv_f_amos**.

Throws `CRivException`—if the helper has a null `CRivClient` object, or if the Client failed to send the query due to an invalid Rendezvous session, or if the query to **riv_f_amos** times out.

See also `CRivRecord`, `CRivClient.getRCTimeOut ()`

2)

Method `public CRivRecord[] getRCHAllModelRecords ()`

Description Return an array of all the model records currently held in the `master.entityByName` table of **riv_model**.

After sending the data, the client engine will expect to receive a response from **riv_model** within the time limit specified by the latency timeout of the client, or the method will throw a `CRivException`.

Note This method is blocking, so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.

Returns An array containing all the model records, as `CRivRecord` objects in the `master.entityByName` table of **riv_model**.

Throws `CRivException`—if the helper has a null `CRivClient` object or if the Client failed to send the query due to an invalid Rendezvous session, or if the query to **riv_model** times out.

See Also `CRivRecord`, `CRivClient.getRCTimeOut ()`

3)

Method	<code>public boolean <i>getRCHAuthPermission</i> (String category, String action)</code>
Description	Return whether the user is allowed to carry out the given action in the specified category. For example, “Can the user clear an Event?”, or “Can the user create new users?”
Parameters	<code>category</code> –the specified category to which the action belongs. <code>action</code> –the action the user wishes to carry out.
Returns	<code>true</code> , if the given action is allowed by the user's Profile in riv_auth . <code>false</code> otherwise.
Throws	<code>java.lang.NullPointerException</code> –if the specified category or action are null. <code>CRivException</code> –if the client is unable to process the query.

4)

Method	<code>public CRivRecord[] <i>getRCHAuthProfiles</i> ()</code>
Description	Return an array of all the profile records currently held in the <code>auth.profiles</code> table of riv_auth . After sending the data, the client engine will expect to receive a response from riv_auth within the time limit specified by the latency timeout of the client, or the method will throw a <code>CRivException</code> . This method is blocking, so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.
Returns	An array containing all the profile records, as <code>CRivRecord</code> objects in the <code>auth.profiles</code> table of riv_auth .
Throws	<code>CRivException</code> –if the helper has a null <code>CRivClient</code> object or if the Client failed to send the query due to an invalid Rendezvous session, or if the query to riv_auth times out.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut ()</code>

5)

Method `public CRivRecord[] getRCHAuthUsers ()`

Description Return an array of all the user records currently held in the `auth.users` table of `riv_auth`.

After sending the data, the client engine will expect to receive a response from **`riv_auth`** within the time limit specified by the latency timeout of the client, or the method will throw a `CRivException`.

This method is blocking so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.

Returns An array containing all the user records, as `CRivRecord` objects in the `auth.users` table of **`riv_auth`**.

Throws `CRivException`—if the helper has a null `CRivClient` object or if the Client failed to send the query due to an invalid Rendezvous session, or if the query to **`riv_auth`** times out.

See Also `CRivRecord`, `CRivClient.getRCTimeOut()`

6)

Method `public CRivRecord getRCHEventRecords(CRivFilter filter)`

Description Get all the event records matching a particular query, expressed as a `CRivFilter`.

After sending the data, the client engine will expect to receive a response from **`riv_f_amos`** within the time limit specified by the latency timeout of the client, or the method will throw an exception.

This method is blocking so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.

Parameters `filter`—the `CRivFilter` query.

Returns An array of all the `CRivRecords` matching the given query.

See Also `CRivFilter`, `CRivRecord`, `CRivClient.getRCTimeOut()`

7)

Method	<code>public CRivRecord[] <i>getRCHEventRecords</i> (CRivVarBindList vbList)</code>
Description	Get all the event records matching a particular query. After sending the data, the client engine will expect to receive a response from <code>riv_f_amos</code> within the time limit specified by the latency timeout of the client, or the method will throw an exception. This method is blocking, so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.
Parameters	<code>vbList</code> —a list of the <code>CRivVarBinds</code> which make up the query. Each name/value pair will form a <code>Name=Value</code> condition. All the name/value pairs in the list will be ANDed together to form the query.
Returns	An array of all the <code>CRivRecords</code> matching the given query.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>

8)

Method	<code>public CRivRecord[] <i>getRCHInterfaces</i> (CRivRecord modelRec)</code>
Description	Return the interfaces for the MODEL record specified. Only the <code>EntityName</code> field needs to be set in <code>modelRec</code> . Returns an array containing the <code>CRivRecord</code> specified and all its interface devices. If the method is called on an <code>CRivRecord</code> which is itself an interface only the same interface is returned. After sending the data, the client engine will expect to receive a response from <code>riv_model</code> within the time limit specified by the latency timeout of the client, or an exception will be thrown. This method is blocking so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.
Parameters	<code>modelRec</code> —the entity name for which the MODEL record is to be returned. Only the <code>EntityName</code> field needs to be set.
Returns	The interfaces for the specified MODEL record.

9)

Method `public CRivRecord[] getRCHModelRecords (CRivFilter filter)`

Description Get all the model records matching a particular query, expressed as a `CRivFilter`.

After sending the data, the client engine will expect to receive a response from **riv_model** within the time limit specified by the latency timeout of the client, or the method will throw an exception.

This method is blocking so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.

Parameters `filter`—the `CRivFilter` query.

Returns An array of all the `CRivRecords` matching the given query.

See Also `CRivFilter`, `CRivRecord`, `CRivClient.getRCTimeOut()`

10)

Method `public CRivRecord[] getRCHModelRecords (CRivVarBindList vbList)`

Description Get all the model records matching a particular query.

After sending the data, the client engine will expect to receive a response from **riv_model** within the time limit specified by the latency timeout of the client, or the method will throw an exception.

This method is blocking so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.

Parameters `vbList`—a list of the `CRivVarBinds` which make up the query. Each name/value pair will form a `Name=Value` condition. All the name/value pairs in the list will be `ANDed` together to form the query.

Returns An array of all the `CRivRecords` matching the given query.

See Also `CRivRecord`, `CRivClient.getRCTimeOut()`

11)

Method	protected byte [] <i>getRCHPacketForAuth</i> (String qryString)
Description	For the given query to riv_auth return a packet of data which is in a suitable format to send to riv_auth . This essentially involves creating a packet of data encapsulating the Username, Password and query.
Parameters	qryString—the query to riv_auth .
Returns	A packet of data as a byte [] encapsulating the query in a suitable format to be read by riv_auth .
Throws	java.lang.NullPointerException—if the query String is null.

12)

Method	protected CRivRecord [] <i>getRCHQueryRecords</i> (String subject, Object qryData)
Description	Generic query method to send the given query into the client, listening on the specified subject for a reply. After sending the data, the client engine will expect to receive a response within the time limit specified by the latency timeout of the client, or the method will throw a CRivException . This method is blocking, so it is recommended that any lengthy queries be performed by placing the call to this method in a separate thread.
Returns	An array containing all the matching records, as CRivRecord objects.
Throws	CRivException —if the helper has a null CRivClient object or if the Client failed to send the query due to an invalid Rendezvous session, or if the query times out.
See Also	CRivRecord

13)

- Method** `public void rchAddAmosListener (IRivRecordListener recListener)`
- Description** Add the specified `IRivRecordListener` to be notified when an update is received from **`riv_f_amos`**.
- Parameters** `recListener`—the `IRivRecordListener` on which to listen for updates to **`riv_f_amos`**.
- Throws** `CRivException`—if the Client session is invalid, or if the specified `IRivRecordListener` has already been added to a listen on a subject.
- See Also** `IRivRecordListener`

14)

- Method** `public void rchAddModelListener (IRivRecordListener recListener)`
- Description** Add the specified `IRivRecordListener` to be notified when an update is received from **`riv_model`**.
- Parameters** `recListener`—the `IRivRecordListener` on which to listen for updates to **`riv_model`**.
- Throws** `CRivException`—if the Client session is invalid, or if the specified `IRivRecordListener` has already been added to a listen on a subject.
- See Also** `IRivRecordListener`

15)

Method	<code>public boolean <i>rchClearEventRecord</i>(CRivRecord eventRec)</code>
Description	<p>Send a request to riv_f_amos to clear the specified record. This method does not check that the user has the required event clearing permission. If event permissions have been set up for profiles in riv_auth then application level code should first call <code>getRCHAuthPermission(String, String)</code> to check that the user of this class has permission to clear events.</p> <p>After sending the data, the client engine will expect to receive a response from riv_f_amos within the time limit specified by the latency timeout of the client, or an exception will be thrown.</p>
Parameters	<code>eventRec</code> —the record to clear. Only the <code>EventId</code> field must be set.
Returns	<code>true</code> , if the Clear request was sent and a response was received from Rendezvous. <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the helper has a null <code>CRivClient</code> object or if the Client failed to send the request due to an invalid Rendezvous session, or if the request to riv_f_amos times out.
See Also	<code>getRCHAuthPermission(String, String)</code> , <code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>

16)

Method	<code>public boolean <i>rchDeleteModelRecord</i>(CRivRecord modelRec)</code>
Description	<p>Send a request to riv_model to delete the specified record from the <code>master.entityByName</code> table.</p> <p>After sending the data, the client engine will expect to receive a response from riv_model within the time limit specified by the latency timeout of the client, or an exception will be thrown.</p>
Parameters	<code>modelRec</code> —the record to delete. Only the <code>ObjectId</code> or <code>EntityName</code> fields need to be set.
Returns	<code>true</code> , if the deletion request was sent and a response was received from Rendezvous. <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the helper has a null <code>CRivClient</code> object or if the Client failed to send the request due to an invalid Rendezvous session, or if the request to riv_model times out.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>

17)

Method	<code>public boolean <i>rchDeleteProfileRecord</i> (CRivRecord profileRec)</code>
Description	Send a request to riv_auth to delete the specified Profile from the <code>auth.profiles</code> table. After sending the data, the client engine will expect to receive a response from riv_auth within the time limit specified by the latency timeout of the client, or an exception will be thrown.
Parameters	<code>profileRec</code> —the Profile record to delete. Only the Profile field needs to be set.
Returns	<code>true</code> , if the deletion request was sent and a response was received from Rendezvous. <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the helper has a null <code>CRivClient</code> object or if the Client failed to send the request due to an invalid Rendezvous session, or if the request to riv_auth times out.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>

18)

Method	<code>public boolean <i>rchDeleteUserRecord</i> (CRivRecord userRec)</code>
Description	Send a request to riv_auth to delete the specified user from the <code>auth.users</code> table. After sending the data, the client engine will expect to receive a response from riv_auth within the time limit specified by the latency timeout of the client, or an exception will be thrown.
Parameters	<code>userRec</code> —the user record to delete. Only the Name field needs to be set.
Returns	<code>true</code> , if the deletion request was sent and a response was received from Rendezvous, <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the helper has a null <code>CRivClient</code> object or if the Client failed to send the request due to an invalid Rendezvous session, or if the request to riv_auth times out.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>

19)

Method	<code>public boolean <i>rchInsertEventRecord</i> (CRivRecord eventRec)</code>
Description	<p>Send the supplied event record to the client engine to be added to the <code>mojo.events</code> table of riv_f_amos. The fields <code>EventId</code>, <code>EntityName</code>, <code>ClassName</code>, <code>Severity</code>, <code>AssignedTo</code>, <code>Acknowledged</code>, <code>EventGroupId</code> and <code>ActionType</code> must all be present in the insert <code>CRivRecord</code>, although they can be zero (integers) or blank <code>Strings</code>.</p> <p>After sending the data, the client engine will expect to receive a response from riv_f_amos within the time limit specified by the latency timeout of the client, or the method will throw an exception.</p>
Parameters	<code>eventRec</code> —the record to insert. This should contain entries for all of the necessary columns in the <code>mojo.events</code> table, with an <code>EventId</code> of 0, which will be filled out by riv_f_amos .
Returns	<code>true</code> , if the insertion request was sent and a response was received from <code>Rendezvous</code> . <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the insert Event <code>CRivRecord</code> did not contain all the mandatory fields, if the helper has a null <code>CRivClient</code> object, if the Client failed to send the request due to an invalid <code>Rendezvous</code> session, or if the request to riv_f_amos times out.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>

20)

Method	<code>public boolean <i>rchInsertModelRecord</i> (CRivRecord modelRec)</code>
Description	<p>Send the supplied model record to the client engine to be added to the <code>master.entityByName</code> table of riv_model.</p> <p>After sending the data, the client engine will expect to receive a response from riv_model within the time limit specified by the latency timeout of the client, or the method will throw an exception.</p>
Parameters	<code>modelRec</code> —the record to insert. This should contain entries for all of the columns in the <code>master.entityByName</code> table, with an <code>ObjectId</code> of 0, which will be filled out by riv_model .
Returns	<code>true</code> , if the insertion request was sent and a response was received from <code>Rendezvous</code> . <code>false</code> otherwise.

Throws CRivException—if the helper has a null CRivClient object, if the record is missing the EntityName field, if the Client failed to send the request due to an invalid Rendezvous session, or if the request to **riv_model** times out.

See Also CRivRecord, CRivClient.getRCTimeOut()

21)

Method public boolean *rchInsertProfileRecord*(CRivRecord profileRec)

Description Send the supplied profile record to the client engine to be added to the auth.profiles table of **riv_auth**. The fields Profile, Rank and Permissions should all be present in the insert CRivRecord, as dictated by the AuthSchema.cfg file in the MWFM installation.

The value of the Permissions field requires particular attention—it should be a LIST type CRivAtom, which is a list of OBJECT type CRivAtoms, which themselves contain OBJECT type CRivAtoms. For instance:

```
Permissions = [ { Users = { pView=1, pChange=0, pCreate=0, pDelete=0 } }
                ]
```

After sending the data, the client engine will expect to receive a response from **riv_auth** within the time limit specified by the latency timeout of the client, or the method will throw an exception.

Parameters profileRec—the record to insert. This should contain entries for all of the necessary columns in the auth.profiles table.

Returns true, if the insertion request was sent and a response was received from Rendezvous. false otherwise.

Throws CRivException—if the insert Profile CRivRecord did not contain all the mandatory fields, if the helper has a null CRivClient object, if the Client failed to send the request due to an invalid Rendezvous session, or if the request to **riv_auth** times out.

See Also CRivAtom, CRivRecord, CRivClient.getRCTimeOut()

22)

Method	<code>public boolean <i>rchInsertUserRecord</i> (CRivRecord userRec)</code>
Description	Send the supplied user record to the client engine to be added to the <code>auth.users</code> table of riv_auth . The fields <code>Name</code> , <code>Password</code> and <code>Profile</code> must all be present in the insert <code>CRivRecord</code> , as dictated by the <code>AuthSchema.cfg</code> file in the MWFM installation. After sending the data, the client engine will expect to receive a response from riv_auth within the time limit specified by the latency timeout of the client, or the method will throw an exception.
Parameters	<code>userRec</code> —the record to insert. This should contain entries for all of the necessary columns in the <code>auth.users</code> table.
Returns	<code>true</code> , if the insertion request was sent and a response was received from <code>Rendezvous</code> . <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the inserted user <code>CRivRecord</code> did not contain all the mandatory fields, if the helper has a null <code>CRivClient</code> object, if the Client failed to send the request due to an invalid <code>Rendezvous</code> session, or if the request to riv_auth times out.
See Also	<code>CRivRecord</code> , <code>CRivClient.getRCTimeOut()</code>


23)

Method	<code>public void <i>rchRemoveAmosListener</i> (IRivRecordListener recListener)</code>
Description	Removes the specified <code>IRivRecordListener</code> so that it no longer receives updates from riv_f_amos . This method performs no function, nor does it throw an exception, if the listener specified by the argument was not previously added as a listener.
See Also	<code>IRivRecordListener</code>

24)

- Method** `public void
rchRemoveModelListener (IRivRecordListener
recListener)`
- Description** Removes the specified `IRivRecordListener` so that it no longer receives updates from **riv_model**. This method performs no function, nor does it throw an exception, if the listener specified by the argument was not previously added as a listener.
- See Also** `IRivRecordListener`

25)

- Method** `public boolean rchUpdateEventRecord (CRivRecord
eventRec, CRivVarBindList evVbList)`
- Description** Send a request to **riv_f_amos** to update the specified record in the `mojo.events` table. The specified `CRivVarBindList` holds the `CRivVarBinds` which give the fields to be modified and the new values they should take.
- After sending the data, the client engine will expect to receive a response from **riv_model** within the time limit specified by the latency timeout of the client, or an exception will be thrown.
-
-  **Caution** In most cases modifying an event record may affect the correlation rules used by `riv_f_amos`, and it may become difficult to track the history of an event through `riv_f_amos`. A valid scenario may be tagging information on to the end of a `Description` field.
-
- Parameters** `eventRec`—the record to modify. Only the `EventId` field needs to be set.
- `evVbList`—the list of modification `CRivVarBinds`. If this list is `null`, or empty, no update will be sent and the method will return `false`. As stated above great caution should be used in deciding which fields to modify.
- Returns** `true`, if the update request was sent and a response was received from `Rendezvous`. `false` otherwise.
- Throws** `CRivException`—if the helper has a `null` `CRivClient` object or if the Client failed to send the request due to an invalid `Rendezvous` session, or if the request to **riv_f_amos** times out.
- See Also** `CRivRecord`, `CRivVarBindList`,
`CRivClient.getRCTimeOut()`

26)

Method	<code>public boolean <i>rchUpdateModelRecord</i> (CRivRecord modelRec, CRivVarBindList modVbList)</code>
Description	<p>Send a request to riv_model to update the specified record from the <code>master.entityByName</code> table. The specified <code>CRivVarBindList</code> holds the <code>CRivVarBinds</code> which give the fields to be modified and the new values they should take.</p> <p>After sending the data, the client engine will expect to receive a response from riv_model within the time limit specified by the latency timeout of the client, or an exception will be thrown.</p>
Parameters	<p><code>modelRec</code>—the record to modify. Only the <code>ObjectId</code> or <code>EntityName</code> fields need to be set, and modifying these two fields should be avoided.</p> <p><code>modVbList</code>—the list of modification <code>CRivVarBinds</code>. If this list is null, or empty, no update will be sent and the method will return false.</p>
Returns	<code>true</code> , if the update request was sent and a response was received from <code>Rendezvous</code> . <code>false</code> otherwise.
Throws	<code>CRivException</code> —if the helper has a null <code>CRivClient</code> object or if the Client failed to send the request due to an invalid <code>Rendezvous</code> session, or if the request to riv_model times out.
See Also	<code>CRivRecord</code> , <code>CRivVarBindList</code> , <code>CRivClient.getRCTimeOut()</code>

27)

Method	<code>public boolean <i>rchUpdateProfileRecord</i> (CRivRecord profileRec, CRivVarBindList modVbList)</code>
Description	<p>Send a request to riv_auth to update the specified <code>Profile</code> record in the <code>auth.profiles</code> table. The specified <code>CRivVarBindList</code> holds the <code>CRivVarBinds</code> which give the fields to be modified and the new values they should take.</p> <p>After sending the data, the client engine will expect to receive a response from riv_auth within the time limit specified by the latency timeout of the client, or an exception will be thrown.</p>
Parameters	<p><code>profileRec</code>—the record to modify. Only the <code>Profile</code> field must be set, and modifying this field should be avoided.</p> <p><code>modVbList</code>—the list of modification <code>CRivVarBinds</code>. If this list is null, or empty, no update will be sent and the method will return false.</p>

Returns `true`, if the update request was sent and a response was received from Rendezvous. `false` otherwise.

Throws `CRivException`—if the helper has a null `CRivClient` object or if the Client failed to send the request due to an invalid Rendezvous session, or if the request to **`riv_auth`** times out.

See Also `CRivRecord`, `CRivVarBindList`, `CRivClient.getRCTimeOut()`

28)

Method `public boolean rchUpdateUserRecord (CRivRecord userRec, CRivVarBindList modVbList)`

Description Send a request to **`riv_auth`** to update the specified user record in the `auth.users` table. The specified `CRivVarBindList` holds the `CRivVarBinds` which give the fields to be modified and the new values they should take.

After sending the data, the client engine will expect to receive a response from **`riv_auth`** within the time limit specified by the latency timeout of the client, or an exception will be thrown.

Parameters `userRec`—the record to modify. Only the Name field must be set, and modifying this field should be avoided.
`modVbList`—the list of modification `CRivVarBinds`. If this list is null, or empty, no update will be sent and the method will return `false`.

Returns `true`, if the update request was sent and a response was received from Rendezvous. `false` otherwise.

Throws `CRivException`—if the helper has a null `CRivClient` object or if the Client failed to send the request due to an invalid Rendezvous session, or if the request to **`riv_auth`** times out.

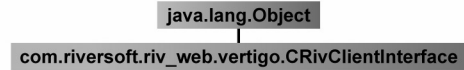
See Also `CRivRecord`, `CRivVarBindList`, `CRivClient.getRCTimeOut()`

Class CRivClientInterface

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-5 Hierarchy of class CRivClientInterface



```
public class CRivClientInterface
extends java.lang.Object
```

Description

This class acts as an interface to a given CRivClient, providing convenience methods to allow polling of an active domain. It abstracts out the Rendezvous methods and objects, acting as an intermediary between the CRivClient object and the user-defined classes.

Inner class summary

Inner Class	Reference Page
1. CRivClientInterface.RCIRecordListener	56
2. CRivClientInterface.RCITimerCallback	57

Constructor summary

Constructor	Reference Page
1. public CRivClientInterface ()	57
2. public CRivClientInterface (CRivClient client)	57

Method summary

Method	Reference Page
1. public CRivClient <i>getRCIClient</i> ()	58
2. public int <i>rciAddOnceOnlyTimer</i> (long interval, IRivTimerCallback timerCallback)	58
3. public int <i>rciAddRepeatTimer</i> (long interval, IRivTimerCallback timerCallback)	58
4. public int <i>rciAddService</i> (String subject, IRivRecordListener userListener)	59
5. public boolean <i>rciRemoveService</i> (IRivRecordListener recListener)	59
6. public boolean <i>rciRemoveTimer</i> (IRivTimerCallback timerCallback)	59
7. public int <i>rciSendToClient</i> (String subject, Object data, IRivRecordListener userListener)	60
8. public int <i>rciTimedSendToClient</i> (String subject, Object data, long timeout, IRivRecordListener userListener)	60
9. public void <i>setRCIClient</i> (CRivClient client)	60

Inner class detail

1)

Inner Class	<code>CRivClientInterface.RCIRecordListener</code>
Description	An inner class implementing the <code>IRivRecordListener</code> interface. This inner class listens for replies on queries, or listener activities, from the <code>CRivClient</code> .

2)

Inner Class	<code>CRivClientInterface.RCITimerCallback</code>
Description	An inner class implementing the Rendezvous RvTimerCallback interface.

Constructor detail

1)

Constructor	<code>public CRivClientInterface ()</code>
Description	Construct a CRivClientInterface. The CRivClient will be null, and until it is set, the CRivClientInterface can perform no useful work.
See Also	<code>setRCIClient (CRivClient)</code>

2)

Constructor	<code>public CRivClientInterface (CRivClient client)</code>
Description	Construct a CRivClientInterface for the specified CRivClient object.
Parameters	<code>client</code> —the client with which this object will interact.

Method detail

1)

Method `public CRivClient getRTCClient()`

Description Return the `CRivClient` object for which this object is acting as an interface.

2)

Method `public int rciAddOnceOnlyTimer(long interval, IRivTimerCallback timerCallback)`

Description Create a new timer activity in the client. This will be a once-only timer which will trigger a single time when its interval elapses.

Parameters `interval`—the timer will trigger when this interval elapses (specified in milliseconds).

`timerCallback`—the object implementing the `IRivTimerCallback` interface will be called on its `rtcTimer()` method when the time interval has elapsed. This should not be null.

Returns An `int`, one of the values defined in `CRivException`, indicating whether the timer was added successfully, e.g., `CRivException.RIV_OK`.

3)

Method `public int rciAddRepeatTimer(long interval, IRivTimerCallback timerCallback)`

Description Create a new timer activity in the client. This will be a repeating timer which will reschedule itself each time it triggers.

Parameters `interval`—the timer will trigger when this interval, specified in milliseconds, elapses.

`timerCallback`—the object implementing the `IRivTimerCallback` interface will be called on its `rtcTimer()` method when the time interval has elapsed. This should not be null.

Returns An `int`, one of the values defined in `CRivException`, indicating whether the repeat timer was added successfully, e.g., `CRivException.RIV_OK`.

4)

Method `public int rciAddService(String subject, IRivRecordListener userListener)`

Description Get the client to add a new listener on the specified broadcast subject.

Parameters `subject`—one of the values from `IRivSubjects`.
`userListener`—the `IRivRecordListener` which will receive any messages on the given broadcast subject. This should not be null.

Returns An `int`, one of the values defined in `CRivException`, indicating whether the service was added successfully, e.g., `CRivException.RIV_OK`.

5)

Method `public boolean rciRemoveService (IRivRecordListener recListener)`

Description Cancel listening interest in the subject for which the specified `IRivRecordListener` is receiving replies.

Parameters `recListener`—the object implementing the `IRivRecord` interface which will receive replies.

Returns `true`, if the specified listener had been previously added as a service, or `false` if the listener had not previously been added, or is null.

6)

Method `public boolean rciRemoveTimer(IRivTimerCallback timerCallback)`

Description Remove the timer activity in the client on which the specified `IRivTimerCallback` is being called.

Parameters `timerCallback`—the object implementing the `IRivTimerCallback` interface.

Returns `true`, if the client had a timer activity associated with the specified `IRivTimerCallback`, `false` otherwise.

Throws `CRivException`—if the `CRivClient` object used by this `CRivClientInterface` is null.

7)

Method	<code>public int rciSendToClient(String subject, Object data, IRivRecordListener userListener)</code>
Description	Send to the client the specified data addressed to the given subject.
Parameters	<p><code>subject</code>—one of the values from <code>IRivSubjects</code>.</p> <p><code>data</code>—the data to send to the client.</p> <p><code>userListener</code>—the <code>IRivRecordListener</code> specified by the user on which to receive replies. This may be <code>null</code> if the user is not interested in the reply from the client.</p>
Returns	An <code>int</code> , one of the values defined in <code>CRivException</code> , indicating whether the data was sent successfully to the client, e.g., <code>CRivException.RIV_OK</code> .

8)

Method	<code>public int rciTimedSendToClient(String subject, Object data, long timeout, IRivRecordListener userListener)</code>
Description	Send to the client the specified data addressed to the given subject. A timed inbox for replies will be created, with a time limit set to the given interval. If no reply is received in this time the <code>IRivRecordListener</code> will be notified through its <code>rriTimeoutReceived()</code> method.
Parameters	<p><code>subject</code>— one of the values from <code>IRivSubjects</code>.</p> <p><code>data</code>—the data to send to the client.</p> <p><code>timeout</code>—the time limit (in milliseconds) in which to receive a reply.</p> <p><code>userListener</code>—the <code>IRivRecordListener</code> specified by the user on which to receive replies. This may be <code>null</code> if the user is not interested in the reply from the Client.</p>
Returns	An <code>int</code> , one of the values defined in <code>CRivException</code> , indicating whether the data was sent successfully to the client, e.g., <code>CRivException.RIV_OK</code> .

9)

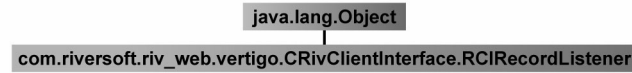
Method	<code>public void setRCIClient(CRivClient client)</code>
Description	Set the <code>CRivClient</code> for which this object is acting as an interface.
Parameters	<code>client</code> —the <code>CRivClient</code> for which this object will act as an interface.

Inner Class `CRivClientInterface.RCIRecordListener`

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Figure 4-6 Hierarchy of class `CRivClientInterface.RCIRecordListener`



```

public class CRivClientInterface.RCIRecordListener
  extends java.lang.Object
  implements IRivRecordListener
  
```

Description

An inner class implementing the `IRivRecordListener` interface which listens for replies on queries, or listener activities, from the `CRivClient`. This acts as an intermediary between the client and the `IRivRecordListener` specified by the user.

Enclosing Class

`CRivClientInterface`

Constructor summary

Constructor	Reference Page
1. <code>public</code> <code>CRivClientInterface.RCIRecordListener</code> (<code>IRivRecordListener</code> <code>userListener</code> , <code>boolean</code> <code>qryType</code>)	62

Method summary

Method	Reference Page
1. <code>public void rrlRivRecordsReceived(CRivRecord[] rivRecs)</code>	62
2. <code>public void rrlTimeOutReceived()</code>	63
3. <code>public void setRCIRLDataHandler(CRivRvDataHandler rvHandler)</code>	63

Constructor detail

1)

Constructor	<code>public CRivClientInterface.RCIRecordListener (IRivRecordListener userListener, boolean qryType)</code>
Description	Construct a new <code>RCIRecordListener</code> to process replies to queries.
Parameters	<code>userListener</code> —the object implementing the <code>IRivRecord</code> interface to send the incoming data. <code>qryType</code> —if <code>true</code> , this listener has been constructed to handle one-off queries.

Method detail

1)

Method	<code>public void rrlRivRecordsReceived(CRivRecord[] rivRecs)</code>
Description	Method will be called when a list of <code>CRivRecords</code> is processed from the transport layer.
Parameters	<code>rivRecs</code> —the list of <code>CRivRecords</code> to be processed.
Specified By	<code>rrlRivRecordsReceived</code> in interface <code>IRivRecordListener</code> .

2)

Method `public void rrlTimeOutReceived()`

Description Method will be called when the time limit has expired for records to be received through the transport layer from Rendezvous.

Specified `rrlTimeOutReceived` in interface
By `IRivRecordListener`.

3)

Method `public void setRCIRLDataHandler(CRivRvDataHandler rvHandler)`

Description Set the Rendezvous callback which passes on its data to this callback.

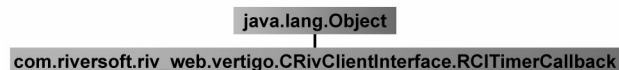
Parameters `rvHandler`—the `CRivRvDataHandler` which is passing its data onto this listener.

Inner Class `CRivClientInterface.RCITimerCallback`

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-7 Hierarchy of class `CRivClientInterface.RCITimerCallback`



```

public class CRivClientInterface.RCITimerCallback
  extends java.lang.Object
  implements COM.TIBCO.rv.RvTimerCallback
  
```

Description

An inner class implementing the Rendezvous `RvTimerCallback` interface. It abstracts out the Rendezvous layer, acting as an intermediary between the client and the user `IRivTimerCallback` object.

Enclosing Class

CRivClientInterface

Constructor summary

Constructor	Reference Page
1. <code>public CRivClientInterface.RCITimerCallback (IRivTimerCallback timerCallback)</code>	64

Method summary

Method	Reference Page
1. <code>public void onTimer (COM.TIBCO.rv.RvTimer invoker)</code>	65

Constructor detail

1)

Constructor	<code>public CRivClientInterface.RCITimerCallback (IRivTimerCallback timerCallback)</code>
Description	Construct a new RCITimerCallback to handle a timer event activity.
Parameter	<code>timerCallback</code> —the object implementing the IRivTimerCallback interface which will be called on its <code>rtcTimer ()</code> method each time its timer is triggered.

Method detail

1)

Method	<code>public void onTimer(COM.TIBCO.rv.RvTimer invoker)</code>
Description	This method will be called by Rendezvous to process each timer event as it occurs. Application-level code should never call this method directly.
Parameters	<code>invoker</code> —each timer activity appoints a handler object to process timer events on its behalf. This parameter receives the timer activity that appointed this handler.
Specified By	<code>onTimer</code> in interface <code>COM.TIBCO.rv.RvTimerCallback</code> .

Class CRivDbEntity

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-8 Hierarchy of class CRivDbEntity



Description

A representation of a database entity, with the general form "database.table.column".

Constructor summary

Constructor	Reference Page
1. <code>public CRivDbEntity(CRivDbEntity dbEntity)</code>	66
2. <code>public CRivDbEntity(String dbName, String tblName, String colName)</code>	66

Method summary

Method	Reference Page
1. <code>public String getRDEColName()</code>	66
2. <code>public String getRDEDbName()</code>	67
3. <code>public String getRDEtblName()</code>	67
4. <code>public String toString()</code>	67

Constructor detail

1)

Constructor	<code>public CRivDbEntity(CRivDbEntity dbEntity)</code>
Description	Construct a new <code>CRivDbEntity</code> by copying the values in the specified <code>CRivDbEntity</code> .
Parameters	<code>dbEntity</code> —the <code>CRivDbEntity</code> to copy.

2)

Constructor	<code>public CRivDbEntity(String dbName, String tblName, String colName)</code>
Description	Construct a new <code>CRivDbEntity</code> with the specified database, table and column names.
Parameters	<code>dbName</code> —the database name. <code>tblName</code> —the table name. <code>colName</code> —the column name.

Method detail

1

Method	<code>public String getRDEColName()</code>
Description	Return the Column name part of the entity.

2)

Method `public String getRDEDbName()`

Description Return the Database name part of the entity.

3)

Method `public String getRDETblName()`

Description Return the Table name part of the entity.

4)

Method `public String toString()`

Description Return a `String` representation of this `CRivDbEntity`, which will be in the format `"Dbname.Tblname.Colname"`. When one or more of these values are null the format will be, `"Dbname" / "Dbname.Tblname" / "Dbname.Colname" / "Dbname" / "Tblname.Colname" / "Tblname" / "Colname"`, depending on which parts are null.

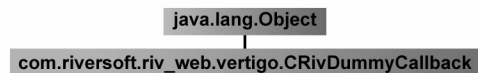
Overrides `toString` in class `java.lang.Object`.

Class CRivDummyCallback

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-9 Hierarchy of class CRivDummyCallback



```

public class CRivDummyCallback
extends java.lang.Object
implements COM.TIBCO.rv.RvDataCallback
  
```

Description

This class is used as a dummy handler to process replies for certain queries. It implements the Rendezvous callback interface `RvDataCallback`, but does not perform any operation in `onData()`. It is used in methods which require a `RvDataCallback` parameter, but where the user is not interested in any replies.

Constructor summary

Constructor	Reference Page
1. <code>public CRivDummyCallback()</code>	68

Method summary

Method	Reference Page
1. <code>public void onData(String subject, COM.TIBCO.rv.RvSender replySender, Object data, COM.TIBCO.rv.RvListener invoker)</code>	69

Constructor detail

1)

Constructor	<code>public CRivDummyCallback()</code>
Description	Construct a <code>CRivDummyCallback</code> .

Method detail

1)

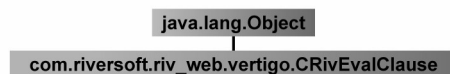
Method	<code>public void onData(String subject, COM.TIBCO.rv.RvSender replySender, Object data, COM.TIBCO.rv.RvListener invoker)</code>
Description	The callback method from the RvDataCallback interface. This performs no function.
Parameters	<p><code>subject</code>—one of the values from IRivSubjects.</p> <p><code>replySender</code>—if the arriving message carries a reply subject, this parameter receives a newly created sender activity that sends to that reply subject. Otherwise, this parameter receives null.</p> <p><code>data</code>—this parameter receives the data object.</p> <p><code>invoker</code>—each timer activity appoints a handler object to process timer events on its behalf. This parameter receives the timer activity that appointed this handler.</p>
Specified By	<code>onData</code> in interface <code>COM.TIBCO.rv.RvDataCallback</code>

Class CRivEvalClause

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-10 Hierarchy of class CRivEvalClause



```
public class CRivEvalClause
extends java.lang.Object
```

Description

This class contains the information dictated by an *eval clause* to be found in active languages such as AOC classes or STORE configs. The **eval clause** allows you to combine multiple evaluation expressions to form component parts of complex algebraic expressions. The **eval clause** itself is the expression that is to be evaluated by the **eval** keyword and coerced into the datatype specified by the cast operator. **eval clause** supports various evaluation expressions, which may be combined algebraically to form one or more complex expressions.

The evaluation expressions may be one, or a combination, of the following:

- A reference to a variable.
- A reference to specific fields of database records known to the current scope.
- A reference to specific fields of database records outside the current operational scope.
- A manipulative statement, which can be used to modify complex data types such as lists or objects, e.g., concatenate two items together, delete items from a list.
- An extraction of an entry within a complex database record using the arrow operator.
- A special expression using the THIS keyword to move in and out of the data Containment Model.
- Algebraic combinations of multiple expressions.

For more information on **eval clauses**, please refer to the appropriate chapter in the RiverSoft NMOS Reference Guide.

Constructor summary

Constructor	Reference Page
1. <code>public CRivEvalClause(CRivEvalClause clauseToCopy)</code>	71
2. <code>public CRivEvalClause(int datatype, String text)</code>	71

Method summary

Method	Reference Page
1. <code>public CRivAtom getRECCompileable()</code>	71
2. <code>public int getRECDatatype()</code>	71
3. <code>public String toString()</code>	72

Constructor detail

1)

Constructor `public CRivEvalClause(CRivEvalClause clauseToCopy)`

Description Construct a clause by creating a deep copy of the supplied clause.

Parameters `clauseToCopy`—the clause to be deep copied.

2)

Constructor `public CRivEvalClause(int datatype, String text)`

Description Construct an `EvalClause` with a data type and a supplied text `String`.

Parameters `datatype`—one of the data type constants defined in `IRivDataType`.
`text`—the text `String` representing the clause, in the appropriate form, such as `"eval (text, '&Description')"`.

Method detail

1)

Method `public CRivAtom getRECCompileable()`

Description Return the compilible text of this `CRivEvalClause`.

2)

Method `public int getRECDatatype()`

Description Return the data type of this `CRivEvalClause`.

Returns One of the data type constants defined in `IRivDataType`.

3)

Method `public String toString()`

Description Return a `String` representation of this `CRivEvalClause`.

Overrides `toString` in class `java.lang.Object`.

Class CRivException

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Figure 4-11 Hierarchy of class `CRivException`



```
public class CRivException
extends java.lang.RuntimeException
```

Description

An exception which signals an error in a MWFM application.

All Implemented Interfaces

`java.io.Serializable`

See Also

Serialized Form

Field summary

Field	Reference Page
1. public static final int <i>CORE_NOT_PRESENT</i>	74
2. public static final int <i>CORRUPT_OBJECT_STORE</i>	74
3. public static final int <i>DUPLICATE_KEY_INSERT</i>	75
4. public static final int <i>DUPLICATE_OBJECT</i>	75
5. public static final int <i>DUPLICATE_PARENT</i>	75
6. public static final int <i>KEY_NOT_FOUND</i>	75
7. public static final int <i>NODE_HAS_ELEMENTAL</i>	75
8. public static final int <i>NODE_HAS_KIDS</i>	76
9. public static final int <i>NODE_MISSING</i>	76
10. public static final int <i>NULL_POINTER_REF</i>	76
11. public static final int <i>PARSE_FAIL</i>	76
12. public static final int <i>RIV_ERROR_BASE</i>	76
13. public static final int <i>RIV_ERROR_MAX</i>	77
14. public static final int <i>RIV_EXIT</i>	77
15. public static final int <i>RIV_FATAL</i>	77
16. public static final int <i>RIV_INFO</i>	77
17. public static final int <i>RIV_OK</i>	78
18. public static final int <i>RIV_SICK</i>	78
19. public static final int <i>ROOT_EMPTY</i>	78
20. public static final int <i>RV_FAILED_TO_SEND</i>	78
21. public static final int <i>RV_INVALID_OUTBOUND_TYPE</i>	78
22. public static final int <i>RV_INVALID_SESSION</i>	79
23. public static final int <i>RV_LISTENER_TIMEOUT</i>	79
24. public static final int <i>SUBTREE_EXCEPTION</i>	79
25. public static final int <i>TYPE_MISMATCH</i>	79
26. public static final int <i>UNDEFINED_ERROR</i>	79
27. public static final int <i>UNRESOLVED_FAULT_RULE</i>	80

Constructor summary

Constructor	Reference Page
1. <code>public CRivException(int errorCode)</code>	80
2. <code>public CRivException(int errorCode, String text)</code>	80

Method summary

Method	Reference Page
1. <code>public String getMessage()</code>	81
2. <code>public int getRECode()</code>	81
3. <code>public int getRESeverity()</code>	81
4. <code>public void rePrintErrorMessage(boolean printStackTrace)</code>	81
5. <code>public void rePrintErrorMessage(boolean printMessage, boolean printStackTrace)</code>	82

Field detail

1)

Field `public static final int CORE_NOT_PRESENT`

Description Indicates that a type of exception has occurred. A record from a database is missing one or more critical fields.

Value -220563

2)

Field `public static final int CORRUPT_OBJECT_STORE`

Description Indicates that a type of exception has occurred. The core database is corrupt. It is probably missing one or more critical records.

Value -220564

3)

Field	<code>public static final int DUPLICATE_KEY_INSERT</code>
Description	Indicates that a type of exception has occurred. A duplicate key has been inserted into a hashtable.
Value	-220560

4)

Field	<code>public static final int DUPLICATE_OBJECT</code>
Description	Indicates that a type of exception has occurred. A duplicate object has been inserted into a hashtable.
Value	-220565

5)

Field	<code>public static final int DUPLICATE_PARENT</code>
Description	Indicates that a type of exception has occurred. Tree corruption indicating that a child has more than one parent.
Value	-220555

6)

Field	<code>public static final int KEY_NOT_FOUND</code>
Description	Indicates that a type of exception has occurred. The search was unsuccessful as the key was not present in the hashtable.
Value	-220561

7)

Field	<code>public static final int NODE_HAS_ELEMENTAL</code>
Description	Indicates that a type of exception has occurred. Invalid operation on a node in a tree which has no children.
Value	-220554

8)

Field `public static final int NODE_HAS_KIDS`

Description Indicates that a type of exception has occurred. Invalid operation on a node in a tree which has children.

Value -220558

9

Field `public static final int NODE_MISSING`

Description Indicates that a type of exception has occurred. A node is missing from a tree.

Value -220557

10)

Field `public static final int NULL_POINTER_REF`

Description Indicates that a type of exception has occurred. Signals an attempt to access a field or method of a null object.

Value -220566

11)

Field `public static final int PARSE_FAIL`

Description Indicates that a type of exception has occurred. Exception indicating that a parse failure occurred due to a syntax error.

Value -220549

12)

Field `public static final int RIV_ERROR_BASE`

Description An internally used constant which is the base error code in a MWFM application. All other errors are defined relative to this value.

Value -220567

13)

Field `public static final int RIV_ERROR_MAX`

Description An internally used constant which specifies the highest error code defined by this class.

Value 100

14)

Field `public static final int RIV_EXIT`

Description An exception that indicates how severe an error in a MWFM application is. This error signals that the user should exit the application.

Value 3

15)

Field `public static final int RIV_FATAL`

Description An exception that signals how severe an error in a MWFM application is. Indicates that a fatal error has occurred. The error will severely disrupt the running of the application and the user should consider exiting.

Value 2

16)

Field `public static final int RIV_INFO`

Description An exception that signals how severe an error in a MWFM application is. Indicates that an exception has occurred which is for informational purposes. The user does not need to exit the application.

Value 0

17)

Field `public static final int RIV_OK`

Description Indicates that everything is OK. This value is often returned by methods throughout the MWFM NMOS Java API to indicate an “OK” status.

Value 1

18)

Field `public static final int RIV_SICK`

Description An exception that signals how severe an error in a MWFM application is. Signals that a severe, but non-fatal, error has occurred in a MWFM application.

Value 1

19)

Field `public static final int ROOT_EMPTY`

Description An object tree is empty, having no root node.

Value -220559

20)

Field `public static final int RV_FAILED_TO_SEND`

Description Failed to send on Rendezvous transport layer.

Value -220552

21)

Field `public static final int RV_INVALID_OUTBOUND_TYPE`

Description Transport layer exception indicating an invalid outbound data type.

Value -220548

22)

Field `public static final int RV_INVALID_SESSION`

Description Exception indicating an invalid Rendezvous session. Rendezvous layer failed to send the data.

Value -220551

23)

Field `public static final int RV_LISTENER_TIMEOUT`

Description Rendezvous layer time out exception, indicating that the waiting time of a timed listener has expired before it has received any data.

Value -220550

24)

Field `public static final int SUBTREE_EXCEPTION`

Description Unexpected subtree exception.

Value -220556

25)

Field `public static final int TYPE_MISMATCH`

Description An attempt has been made to carry out an invalid operation on an object which is not of the correct type.

Value -220562

26)

Field `public static final int UNDEFINED_ERROR`

Description Undefined error code.

Value -220567

27)

Field `public static final int UNRESOLVED_FAULT_RULE`

Description Unresolved fault rule reference in active object language.

Value -220553

Constructor detail

1)

Constructor `public CRivException(int errorCode)`Description Construct a `CRivException` to signify an exception with the specified error code, one of the constants defined by this class.Parameter `errorCode`—the specified error code, one of the constants defined by this class. If the value is not one of the error code constants defined by this class, the error code will be set to `UNDEFINED_ERROR`.

2)

Constructor `public CRivException(int errorCode, String text)`Description Construct a `CRivException` to signify an exception with the specified error code, one of the constants defined by this class. Additionally, append the specified text to the error `String` reported by this exception.Parameter `errorCode`—the specified error code, one of the constants defined by this class. If the value is not one of the error code constants defined by this class, the error code will be set to `UNDEFINED_ERROR`.
`text`—the text to be appended to the standard error `String`.

Method detail

1)

Method `public String getMessage()`

Description Return the error message `String` for this exception.

Overrides `getMessage` in class `java.lang.Throwable`

2)

Method `public int getRECode()`

Description Return the error code for this exception, which identifies what type of exception has occurred.

Returns One of the error code constants defined by this class.

3)

Method `public int getRESeverity()`

Description Return the severity of this exception.

Returns One of the severity constants defined by this class.

See Also `RIV_INFO`, `RIV_SICK`, `RIV_FATAL`, `RIV_EXIT`

4)

Method `public void rePrintErrorMessage(boolean printStackTrace)`

Description Print the error message for this exception to the standard error stream.

Parameters `printStackTrace`—if `true`, prints the backtrace.

5)

Method `public void rePrintErrorMessage(boolean printMessage, boolean printStackTrace)`

Description Allows printing, or not, of the error message and stack trace for this exception to the standard error stream.

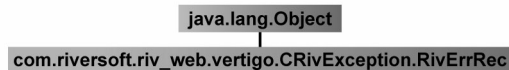
Parameters `printMessage`—if true, prints the error message.
`printStackTrace`—if true, prints the backtrace.

Inner Class `CRivException.RivErrRec`

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-12 Hierarchy of class `CRivException.RivErrRec`



```

public static class CRivException.RivErrRec
  extends java.lang.Object
  
```

Description

Internal wrapper class for the error code, level and message.



Note

`CRivException.RivErrRec`—Application-level code should never need to use this class directly.

Enclosing Class

`CRivException`

Constructor summary

Constructor	Reference Page
1. <code>public CRivException.RivErrRec(int code, int level, String errorStr)</code>	83

Method summary

Method	Reference Page
1. <code>public int getERCode()</code>	84
2. <code>public int getERLevel()</code>	84
3. <code>public String getERString()</code>	84

Constructor detail

1)

Constructor	<code>public CRivException.RivErrRec(int code, int level, String errorStr)</code>
Description	Construct the <code>RivErrRec</code> given the error code, severity level and error message <code>errorStr</code> .
Parameters	<p><code>code</code>—the specified error code, one of the constants defined by this class.</p> <p><code>level</code>—the severity level of this error: <code>RIV_INFO</code>, <code>RIV_SICK</code>, <code>RIV_FATAL</code>, <code>RIV_EXIT</code>.</p> <p><code>errorStr</code>—the standard error message for this exception.</p>

Method detail

1)

Method `public int getERCode()`

Description Return the error code for this error.

Returns The code for this error, which will be one of the constants defined in `CRivException`.

2)

Method `public int getERLevel()`

Description Return the severity level of this error.

Returns One of the constants defined by this class, i.e., `RIV_INFO`, `RIV_SICK`, `RIV_FATAL` or `RIV_EXIT`.

3)

Method `public String getERString()`

Description Return the message `String` associated with this error record.

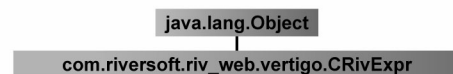
Returns The message `String` for this error.

Class CRivExpr

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-13 Hierarchy of class *CRivExpr*



```

public class CRivExpr
extends java.lang.Object
  
```

Description

This class contains and evaluates expressions which are used as part of boolean query arithmetic. For example, the following are all valid expressions:

```
table.col
(12*(3+4))
-3
'a piece of text'
```

These can be constructed from `CRivAtom`, `CRivDbEntity` or `CRivEvalClause` objects, and are used when constructing `CRivQueryAtoms` for `CRivFilters`. A `CRivQueryAtom` will take the form of `CRivExpr int CRivExpr` e.g., `'ClassName' <> 'MyRouter'`, where `ClassName` and `MyRouter` are both wrapped inside `CRivExpr` objects, and `int` is one of the operator constants defined in `IRivOper`.

Only `CRivExprs` which are of type `SUPPLIED` are fully utilized by `CRivQueryAtom`. The other types will be expanded upon in later releases of the NMOS Java API.

See Also

`CRivAtom`, `CRivDbEntity`, `CRivEvalClause`, `CRivQueryAtom`, `CRivFilter`, `IRivOper`

Field summary

Field	Reference Page
1. <code>public static final int CLAUSE</code>	86
2. <code>public static final int IGNORED</code>	87
3. <code>public static final int INTERNAL</code>	87
4. <code>public static final int SUPPLIED</code>	87

Constructor summary

Constructor	Reference Page
1. <code>public CRivExpr()</code>	87
2. <code>public CRivExpr(CRivAtom atmVal)</code>	88
3. <code>public CRivExpr(CRivDbEntity dbEntVal)</code>	88
4. <code>public CRivExpr(CRivEvalClause clauseVal)</code>	88
5. <code>public CRivExpr(CRivExpr exprVal)</code>	89

Method summary

Method	Reference Page
1. <code>public CRivAtom getREAtomVal()</code>	89
2. <code>public CRivEvalClause getREClauseVal()</code>	89
3. <code>public CRivDbEntity getREEntityVal()</code>	89
4. <code>public int getREType()</code>	89
5. <code>public String toString()</code>	90

Field detail

1)

Field	<code>public static final int CLAUSE</code>
Description	Type constant indicating that this <code>CRivExpr</code> is constructed from a <code>CRivEvalClause</code> .
Value	2
See Also	<code>CRivEvalClause</code>

2)

Field	<code>public static final int IGNORED</code>
Description	Type constant representing an empty <code>CRivExpr</code> , which is ignored.
Value	3

3)

Field	<code>public static final int INTERNAL</code>
Description	Type constant indicating that this <code>CRivExpr</code> is constructed from a <code>CRivDbEntity</code> .
Value	1
See Also	<code>CRivDbEntity</code>

4)

Field	<code>public static final int SUPPLIED</code>
Description	Type constant indicating that this <code>CRivExpr</code> is constructed from a supplied <code>CRivAtom</code> .
Value	0
See Also	<code>CRivAtom</code>

Constructor detail

1)

Constructor	<code>public CRivExpr()</code>
Description	Construct an empty <code>CRivExpr</code> , whose datatype will be <code>IGNORED</code> .

2)

Constructor	<code>public CRivExpr(CRivAtom atmVal)</code>
Description	Construct an expression from a supplied <code>CRivAtom</code> value. The datatype will be <code>SUPPLIED</code> .
Parameter	<code>atmVal</code> —the <code>CRivAtom</code> value for ‘this’ <code>CRivExpr</code> .
See Also	<code>CRivAtom</code>

3)

Constructor	<code>public CRivExpr(CRivDbEntity dbEntVal)</code>
Description	Construct an expression from a supplied <code>CRivDbEntity</code> value. The datatype will be <code>INTERNAL</code> .
Parameter	<code>dbEntVal</code> —the <code>CRivDbEntity</code> value for ‘this’ <code>CRivExpr</code> .
See Also	<code>CRivDbEntity</code>

4)

Constructor	<code>public CRivExpr(CRivEvalClause clauseVal)</code>
Description	Construct an expression from a supplied <code>CRivEvalClause</code> value. The datatype will be <code>CLAUSE</code> .
Parameter	<code>clauseVal</code> —the <code>CRivEvalClause</code> value for ‘this’ <code>CRivExpr</code> .
See Also	<code>CRivEvalClause</code>

5)

Constructor	<code>public CRivExpr(CRivExpr exprVal)</code>
Description	Construct an expression by creating a deep copy of a supplied value.
Parameter	<code>exprVal</code> —the <code>CRivExpr</code> to be deep copied.

Method detail

1)

Method	<code>public CRivAtom getREAtomVal()</code>
Description	Return the value of the expression as an atom.
Returns	The compilable form of the <code>CRivEvalClause</code> if the type is <code>CLAUSE</code> , or <code>null</code> if the type is <code>INTERNAL</code> or <code>IGNORED</code> .
See Also	<code>CRivEvalClause.getRECCompileable()</code>

2)

Method	<code>public CRivEvalClause getREClauseVal()</code>
Description	Return the clause value of this expression.
Returns	<code>null</code> , if this expression is not of type <code>CLAUSE</code> .

3)

Method	<code>public CRivDbEntity getREEntityVal()</code>
Description	Return the entity value of this expression.
Returns	<code>null</code> , if this expression is not of type <code>INTERNAL</code> .

4)

Method	<code>public int getREType()</code>
Description	Return the type of the expression.
Returns	One of the constant <code>ints</code> defined in this class.

5)

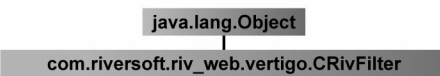
Method `public String toString()`Description Return a `String` representation of this `CRivExpr`.Overrides `toString` in class `java.lang.Object`

Class CRivFilter

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-14 Hierarchy of class CRivFilter



```

public class CRivFilter
  extends java.lang.Object
  implements IRivAlgebraic
  
```

Description

This class stores, in a tree format, queries such as $((y < 3) \text{ AND } (x = 1)) \text{ OR } (z \neq 0)$. Each node can be a left right `CRivFilter` subtree plus algebraic operator, or two `CRivQueryAtoms` (e.g., ‘ $y < 3$ ’ is a single `CRivQueryAtom`) and algebraic combination. If the `CRivFilter` consists of a single `CRivQueryAtom`, the query atom is referred to as the ‘kernel’ part of the filter.

The method `rfEvalFilter(CRivRecord)` is used to determine whether a given `CRivRecord` passes the filter and is built to be evaluated recursively.

A `CRivFilter` may be constructed by parsing from a `String` in the appropriate format, by using filter parsers such as `CRivFilterParser`. Alternatively it may be built by combining two other `CRivFilter` objects with an algebraic operator, or be based around a single `CRivQueryAtom`.

See Also

`CRivFilterParser`, `CRivQueryAtom`, `CRivRecord`, `IRivAlgebraic`

Constructor summary

Constructor	Reference Page
1. <code>public CRivFilter()</code>	92
2. <code>public CRivFilter(CRivFilter filterToCopy)</code>	92
3. <code>public CRivFilter(CRivFilter left, CRivFilter right, int oper)</code>	92
4. <code>public CRivFilter(CRivQueryAtom qryAtm)</code>	93

Method summary

Method	Reference Page
1. <code>public CRivQueryAtom getRFKernel()</code>	93
2. <code>public CRivFilter getRFLeftSub()</code>	93
3. <code>public int getRFOperator()</code>	93
4. <code>public CRivFilter getRFRightSub()</code>	94
5. <code>public boolean rfAddLeftBranch(CRivFilter leftFilter)</code>	94
6. <code>public boolean rfAddRightBranch(CRivFilter rightFilter)</code>	94
7. <code>public boolean rfEvalFilter(CRivRecord record)</code>	94
8. <code>public static String rfPrint(CRivFilter filter)</code>	95
9. <code>public boolean rfReplaceBranch(CRivFilter oldBranch, CRivFilter newBranch)</code>	95
10. <code>public void setRFOperator(int oper)</code>	95
11. <code>public String toString()</code>	95

Constructor detail

1)

Constructor	<code>public CRivFilter()</code>
Description	Construct an empty <code>CRivFilter</code> . The algebraic operator will be the default <code>RA_NO_OP</code> .
See Also	<code>IRivAlgebraic.RA_NO_OP</code>

2)

Constructor	<code>public CRivFilter(CRivFilter filterToCopy)</code>
Description	Construct a <code>CRivFilter</code> by creating a deep copy of the given <code>CRivFilter</code> . Each branch, or leaf, of the tree will be copied recursively.
Parameters	<code>filterToCopy</code> —the <code>CRivFilter</code> to copy. If this is null, a new empty <code>CRivFilter</code> object will be constructed.

3)

Constructor	<code>public CRivFilter(CRivFilter left, CRivFilter right, int oper)</code>
Description	Construct a new <code>CRivFilter</code> from the left and right subtree <code>CRivFilter</code> objects and the given algebraic operator.
Parameters	<p><code>left</code>—the left subtree from which to construct this new <code>CRivFilter</code>.</p> <p><code>right</code>—the right subtree from which to construct this new <code>CRivFilter</code>.</p> <p><code>oper</code>—one of the constants defined in <code>IRivAlgebraic</code>.</p>

4)

Constructor	<code>public CRivFilter(CRivQueryAtom qryAtm)</code>
Description	Construct a new <code>CRivFilter</code> from the single specified <code>CRivQueryAtom</code> .
Parameters	<code>qryAtm</code> —the <code>CRivQueryAtom</code> value of this <code>Filter</code> .
See Also	<code>CRivQueryAtom</code>

Method detail

1)

Method	<code>public CRivQueryAtom getRFKernel()</code>
Description	Return the kernel part of this <code>CRivFilter</code> , if it exists. This will be non-null for simple <code>CRivFilters</code> which were constructed from a single <code>CRivQueryAtom</code> to perform "elemental" queries such as "EventId > 10".
Returns	null, if this <code>CRivFilter</code> contains left of right sub-trees.
See Also	<code>CRivFilter(CRivQueryAtom)</code>

2)

Method	<code>public CRivFilter getRFLeftSub()</code>
Description	Return the left hand subtree of this <code>CRivFilter</code> , if it exists.
Returns	null, if this <code>CRivFilter</code> is empty, or is a simple <code>CRivFilter</code> constructed from a single <code>CRivQueryAtom</code> .

3)

Method	<code>public int getRFOperator()</code>
Description	Return the operator which combines the left- and right-hand sub-trees of the filter.
Returns	One of the constants defined in <code>IRivAlgebraic</code>

4)

Method `public CRivFilter getRRRightSub()`

Description Return the right hand sub-tree of this `CRivFilter`, if it exists.

Returns `null`, if this `CRivFilter` is empty, or is a simple `CRivFilter` constructed from a single `CRivQueryAtom`.

5)

Method `public boolean rfAddRightBranch(CRivFilter rightFilter)`

Description Add in a right branch to the filter.

Parameters `rightFilter`—the right subtree to add to ‘this’.

Returns `true`, if the branch was added successfully, `false` if the branch was not added i.e., if there was already a right branch, or if the kernel `CRivQueryAtom` is not `null`.

6)

Method `public boolean rfAddLeftBranch(CRivFilter leftFilter)`

Description Add in a left branch to the filter.

Parameters `leftFilter`—the left subtree to add to ‘this’.

Returns `true`, if the branch was added successfully, `false` if the branch was not added, i.e., if there was already a left branch, or if the kernel `CRivQueryAtom` is not `null`.

7)

Method `public boolean rfEvalFilter(CRivRecord record)`

Description Evaluate whether the given `CRivRecord` passes the condition specified in this `CRivFilter`.

Parameters `record`—the `CRivRecord` to pass through this filter.

Returns `true`, if the `CRivRecord` passes the query, `false` otherwise.

Throws `java.lang.NullPointerException`—if the specified `CRivRecord` is `null`.

8)

Method `public static String rfPrint(CRivFilter filter)`

Description Class method used to return a `String` representation of the given `CRivFilter`. The primary purpose of this method is to recursively print all the branches in a filter.

Parameters `filter`—the `CRivFilter` to convert to a `String` representation.

9)

Method `public boolean rfReplaceBranch(CRivFilter oldBranch, CRivFilter newBranch)`

Description Replace an existing branch in this filter tree with a new branch. This method will search recursively through the tree structure of this filter to look for the given old branch.

Parameters `oldBranch`—the old branch to change.
`newBranch`—the branch to substitute for the old one.

Returns `true`, if the given old branch is successfully found in this filter and is replaced with the specified new branch.

Throws `CRivException`—if the old or new branch are `null`.

10)

Method `public void setRFOperator(int oper)`

Description Set the operator which combines the left- and right-hand sub-trees of the filter.

Parameters `oper`—one of the constants defined in `IRivAlgebraic`

11)

Method `public String toString()`

Description Return a `String` representation of this `CRivFilter`.

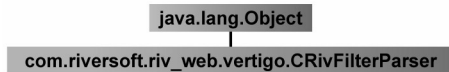
Overrides `toString` in class `java.lang.Object`

Class CRivFilterParser

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-15 Hierarchy of class CRivFilterParser



```
public class CRivFilterParser
extends java.lang.Object
```

Description

This class defines an ultimate wrapper for the Filter parser, *RivFilter*, allowing a text *String* in the appropriate format to be parsed to a MWFM filter object, *CRivFilter*. The class *RivFilter* should never be used directly, but access should always be through this class, by calling the `rfpParse (String text)` method which will return a *CRivFilter* which has been parsed from the given text *String*.

This class is commonly used to process filter objects which arrive from MWFM data engines via OQL requests contained in *CRivAtom* objects of type `IRivDataType.RDT_STRING`.

CRivFilterParser also provides several class methods for converting operators from *IRivOper* and *IRivAlgebraic* to *Strings*.

See Also

CRivFilter, *IRivAlgebraic*, *IRivOper*

Constructor summary

Constructor	Reference Page
1. <code>public CRivFilterParser()</code>	97

Method summary

Method	Reference Page
1. <code>public static int rfpAlgebraicToInt(String operString)</code>	98
2. <code>public static String rfpAlgebraicToString(int algebraic)</code>	98
3. <code>public static int rfpOperatorToInt(String operString)</code>	98
4. <code>public static String rfpOperatorToString(int oper)</code>	99
5. <code>public CRivFilter rfpParse(String text)</code>	99

Constructor detail

1)

Constructor	<code>public CRivFilterParser()</code>
Description	Construct a new <code>CRivFilterParser</code> object to parse text Strings to <code>CRivFilter</code> objects.
See Also	<code>CRivFilter</code>

Method detail

1)

Method `public static int rfpAlgebraicToInt(String operString)`

Description Convert an algebraic operator from a `String` to an `int`, i.e., "And" to `IRivAlgebraic.RA_AND`; "Or" to `IRivAlgebraic.RA_OR`; "Not" to `IRivAlgebraic.RA_NOT`; "NoOp" to `IRivAlgebraic.RA_NO_OP`. This method uses a case insensitive comparison of `Strings`.

Parameters `operString`—the algebraic operator `String` to be converted.

Returns One of the constants defined in `IRivAlgebraic`, or -1, if the `String` did not match any of these operators.

See Also `IRivAlgebraic`

2)

Method `public static String rfpAlgebraicToString (int algebraic)`

Description Convert an algebraic operator to a `String`. For example, `IRivAlgebraic.RA_AND` is converted to the `String` "And".

Parameters `algebraic`—one of the algebraic operator constants defined in `IRivAlgebraic` (`RA_AND`, `RA_OR`, `RA_NO_OP`, `RA_NOT`).

Returns The algebraic operator converted to a `String`.

See Also `IRivAlgebraic`

3)

Method `public static int rfpOperatorToInt(String operString)`

Description Convert a `String` to an `int` operator. For example "<>" is converted to `IRivOper.RO_NOT_EQUALS`. The method uses a case insensitive comparison of `Strings`.

Returns One of the constants defined in `IRivOper`, or -1, if the `String` did not match any of these operators.

Parameters `operString`—the operator `String` to be converted.

See Also `IRivOper`

4)

Method `public static String rfpOperatorToString(int oper)`

Description Convert an operator to a `String`. For example `IRivOper.RO_NOT_EQUALS` is converted to "<>".

Parameters `oper`—one of the operator constants defined in `IRivOper`.

Returns A `String` representation of the given operator, or an empty `String` if the operator is not one of the values defined in `IRivOper`.

See Also `IRivOper`

5)

Method `public CRivFilter rfpParse(String text)`

Description Parse a text `String` into a `CRivFilter` object.

Returns The `CRivFilter` parsed from the given `String`, or `null` if the text passed in was `null` or of zero length.

Parameters `text`—the text to be parsed into a `CRivFilter`.

Throws `ParseException`—if the text `String` was not of the correct format to be successfully parsed to a `CRivFilter`.

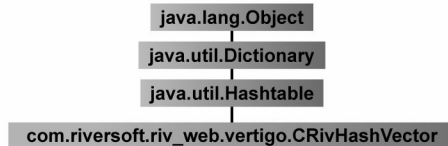
See Also `CRivFilter`

Class CRivHashVector

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-16 Hierarchy of class CRivHashVector



```
public class CRivHashVector
extends java.util.Hashtable
```

Description

This class is an extension of `java.util.Hashtable`, which allows values to be extracted in the order in which they were added. As in a `Hashtable`, the class implements a data structure in which `key` objects are mapped to `value` objects, allowing the efficient extraction of the value associated with a given key. Unlike a `Hashtable`, the data structure additionally keeps a record of the order in which the `key/value` pairs were added.

The table only supports unique keys. If a duplicate key is added, it simply overwrites the existing value at the same index as the first value (unless `rhvPutAt()` is used).

All Implemented Interfaces

`java.lang.Cloneable`, `java.util.Map`, `java.io.Serializable`

See Also

Serialized Form

Constructor summary

Constructor	Reference Page
1. <code>public CRivHashVector()</code>	102
2. <code>public CRivHashVector(int initialCapacity)</code>	102

Method summary

Method	Reference Page
1. public void <i>clear</i> ()	102
2. public Object <i>clone</i> ()	102
3. public boolean <i>containsKey</i> (Object key)	103
4. public Object <i>forcePut</i> (Object key, Object value)	103
5. public Object <i>get</i> (Object key)	104
6. public int <i>getRHVFirstIndexOfValue</i> (Object value)	104
7. public int <i>getRHVIndexOfKey</i> (Object key)	104
8. public Object <i>getRHVKeyAt</i> (int index)	105
9. public Object <i>getRHVValueAt</i> (int index)	105
10. public Object <i>put</i> (Object key, Object value)	105
11. public void <i>putAll</i> (java.util.Map t)	106
12. public Object <i>remove</i> (Object key)	106
13. public Object <i>rhvPutAt</i> (Object key, Object value, int index)	106

Constructor detail

1)

Constructor	<code>public CRivHashVector()</code>
Description	Constructs an empty <code>CRivHashVector</code> so that its internal data array has size 10 and its standard capacity increment is zero.

2)

Constructor	<code>public CRivHashVector(int initialCapacity)</code>
Description	Constructs an empty <code>CRivHashVector</code> with an internal data array of the specified initial size and a standard capacity increment of zero.
Parameters	<code>initialCapacity</code> —the initial capacity of the <code>CRivHashVector</code> .

Method detail

1)

Method	<code>public void clear()</code>
Description	Clears this <code>CRivHashVector</code> so that it contains no keys.
Overrides	<code>clear</code> in class <code>java.util.Hashtable</code>

2)

Method	<code>public Object clone()</code>
Description	Creates a shallow copy of this <code>CRivHashVector</code> . All the structure of the <code>CRivHashVector</code> itself is copied, but the keys and values are not cloned. This is a relatively expensive operation.
Overrides	<code>clone</code> in class <code>java.util.Hashtable</code> .
Returns	A clone of the hashtable.

3)

Method `public boolean containsKey(Object key)`

Description Tests if the specified object is a key in this hashtable.

Overrides `containsKey` in class `java.util.Hashtable`

Parameters `key`—a valid key.

Returns `true` if, and only if, the specified object is a key in this hashtable, as determined by the `equals` method. `false` otherwise.

See Also `Hashtable.contains(Object)`

4)

Method `public Object forcePut(Object key, Object value)`

Description Maps the specified key to the specified value in this `CRivHashVector`. If the `CRivHashVector` already contains a value for this key, it will replace the existing value with the new specified value. Neither the key nor the value can be `null`.

The value can be retrieved by calling the `get(Object key)` method with a key that is equal to the original key.

An internal `Vector` holds the keys, so that values can be obtained from the `CRivHashVector` in the order in which they were added.

Parameters `key`—the hashtable key.
`value`—the value.

Returns The existing value of the specified key in this `CRivHashVector` if the `CRivHashVector` already contains this key, or `null` if it did not contain this key.

Throws `java.lang.NullPointerException`—if the key or value is `null`.

See Also `Put(Object, Object)`, `Object.equals(Object)`,
`get(Object)`

5)

Method `public Object get(Object key)`

Description Returns the value to which the specified key is mapped in this hashtable. Overrides `java.util.Hashtable get ()` so that it simply uses the `equals ()` method of the `Object` key to look up the specified key in the `CRivHashVector`.

Overrides `get` in class `java.util.Hashtable`.

Parameters `key`—a key in the hashtable.

Returns The value to which the key is mapped in this hashtable; `null` if the key is not mapped to any value in this hashtable.

See Also `put (Object, Object)`

6)

Method `public int getRHVFirstIndexOfValue(Object value)`

Description Searches for the first occurrence of the given value, beginning the search at index 0, and testing for equality using the `equals` method of the `Object` value.

Parameters `value`—the desired value.

Returns The index of the first occurrence of the value in this `CRivHashVector`; returns -1 if the key is not found.

7)

Method `public int getRHVIndexOfKey(Object key)`

Description Searches for the first occurrence of the given key and tests for equality using the `equals` method of the `Object` key.

Parameters `key`—the desired key.

Returns The index of the first occurrence of the key in this `CRivHashVector`; returns -1 if the key is not found.

8)

Method `public Object getRHVKeyAt(int index)`

Description Returns the key at the specified index.

Parameters `index`—an index into this `CRivHashVector`.

Returns The key at the specified index.

Throws `ArrayIndexOutOfBoundsException`—if the index is negative or not less than the current size of this `CRivHashVector` object.

9)

Method `public Object getRHVValueAt(int index)`

Description Returns the value at the specified index.

Parameters `index`—an index into this `CRivHashVector`.

Returns The value at the specified index.

Throws `ArrayIndexOutOfBoundsException`—if the index is negative or not less than the current size of this `CRivHashVector` object.

10)

Method `public Object put(Object key, Object value)`

Description Maps the specified key to the specified value in this `CRivHashVector` only if this `CRivHashVector` does not already contain this key. To force addition of a value for a duplicate key, use `forcePut()`. Neither the key nor the value can be `null`. The value can be retrieved by calling the `get` method with a key that is equal to the original key.

An internal `Vector` holds the keys, so that values can be obtained from the `CRivHashVector` in the order in which they were added.

Overrides `put` in class `java.util.Hashtable`

Parameters `key`—the hashtable key.
`value`—the value.

Returns The existing value of the specified key in this `CRivHashVector` if the `CRivHashVector` already contains this key, or `null` if it did not contain this key.

Throws `java.lang.NullPointerException`—if the key or value is null.

See Also `forcePut (Object, Object)`
`Object.equals (Object), get (Object)`

11)

Method `public void putAll(java.util.Map t)`

Description Copies all of the mappings from the specified Map to this Hashtable. These mappings will replace any mappings that this Hashtable had for any of the keys currently in the specified Map.

Parameters `t`—mappings to be stored in this CRivHashVector.

Overrides `putAll` in class `java.util.Hashtable`

12)

Method `public Object remove(Object key)`

Description Removes the key (and its corresponding value) from this CRivHashVector. This method does nothing if the key is not in the hashtable.

Overrides `remove` in class `java.util.Hashtable`

Parameters `key`—the key that needs to be removed.

Returns The value to which the key had been mapped in this hashtable, or null if the key did not have a mapping.

13)

Method `public Object rhvPutAt(Object key, Object value, int index)`

Description Maps the specified key to the specified value in this hashtable. Neither the key nor the value can be null.

The value can be retrieved by calling the `get` method with a key that is equal to the original key.

An internal `Vector` holds the keys, so that values can be obtained from the `CRivHashVector` in the order in which they were added. The new key is placed at the specified index.

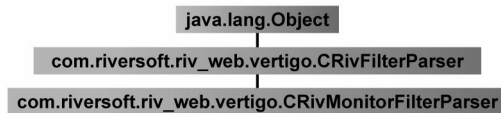
Parameters	key—the hashtable key. value—the value. index—the position at which to insert the key.
Returns	The previous value of the specified key in this hashtable, or null if it did not have one.
Throws	java.lang.NullPointerException—if the key or value is null. ArrayIndexOutOfBoundsException—if the index is not valid.
See Also	Object.equals(Object), get(Object)

Class CRivMonitorFilterParser

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-17 Hierarchy of class CRivMonitorFilterParser



```
public class CRivMonitorFilterParser
extends CRivFilterParser
```

Description

An extension of CRivFilterParser which allows the name fields in filters to include the style of addressing "Name[.x]+", i.e., a name, followed by one or more ".x" where x is any number, e.g., "ifOperStatus.1". This style of addressing would not be parsed successfully by CRivFilterParser. This is commonly encountered in defining Threshold filters in polling agents, particularly SNMP agents, used by riv_monitor.

See Also

CRivFilterParser

Constructor summary

Constructor	Reference Page
1. <code>public CRivMonitorFilterParser()</code>	108

Method summary

Method	Reference Page
1. <code>public CRivFilter rfpParse(String text)</code>	108

Constructor detail

1)

Constructor `public CRivMonitorFilterParser()`

Description Construct a new `CRivMonitorFilterParser` object to parse text `String`s to `CRivFilter` objects allowing the name fields in filters to include the style of addressing "Name [.x] +".

See Also `CRivFilter`

Method detail

1)

Method `public CRivFilter rfpParse(String text)`

Description Parse a text `String` into a `CRivFilter` object.

Parameters `text`—the text to be parsed into a `CRivFilter`.

Overrides `rfpParse` in class `CRivFilterParser`.

Returns The `CRivFilter` parsed from the given `String`, or null if the text passed in was of zero length.

Throws `ParseException`—if the text `String` was not of the correct format to be successfully parsed to a `CRivFilter`.

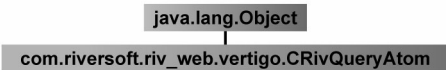
See Also `CRivFilter`

Class CRivQueryAtom

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-18 Hierarchy of class CRivQueryAtom



```

public class CRivQueryAtom
extends java.lang.Object
  
```

Description

This class encodes the basic query on the data store. It carries a query of the sort: “leftHandName relOp rightHandValue”, where leftHandName and rightHandValue are ultimately CRivExpr objects. A query atom may also be constructed from CRivAtom objects for convenience, and it should be noted in this release that only queries using CRivExpr objects of type CRivExpr.SUPPLIED will be fully evaluated. Other types of CRivExpr will be converted to CRivAtom objects of type IRivDataType.RDT_STRING.

relOp is one of the relative operator constants ==, !=, <, <=, >, >=, In and Not In, as defined in IRivOper. EventId>100 and ClassName == 'Router' are examples of CRivQueryAtom objects.

In a future release, all the operators defined in IRivOper will be supported, so that Matches, Not Matches, Exists, Between, Is null, and Is Not null tests can also be evaluated.

The class is used to determine whether a CRivRecord satisfies a query by calling the method rqaEvalQuery(CRivRecord record), and is usually used in conjunction with the class CRivFilter.

See Also

CRivExpr, CRivAtom, IRivOper, CRivRecord, CRivFilter

Constructor summary

Constructor	Reference Page
1. public CRivQueryAtom()	111
2. public CRivQueryAtom(CRivAtom name, int relOp, CRivAtom value)	111

Constructor	Reference Page
3. public CRivQueryAtom (CRivExpr lhExpr, int relOp, CRivExpr rhExpr)	111
4. public CRivQueryAtom (CRivQueryAtom qryAtomToCopy)	112

Method summary

Method	Reference Page
1. public CRivAtom <i>getRQALeft</i> ()	112
2. public int <i>getRQAOperator</i> ()	112
3. public CRivAtom getRQARight ()	112
4. public int <i>getRQAValueType</i> ()	112
5. public boolean <i>rqaEvalQuery</i> (CRivRecord record)	113
6. public void setRQALeft (CRivAtom name)	113
7. public void <i>setRQAOperator</i> (int oper)	113
8. public void setRQARight (CRivAtom value)	113
9. public String <i>toString</i> ()	114

Constructor detail

1)

Constructor	<code>public CRivQueryAtom()</code>
Description	Construct an empty <code>CRivQueryAtom</code> , where the left- and right-hand sides are null, and the relative operative is <code>IRivOper.RO_NONE</code>

2)

Constructor	<code>public CRivQueryAtom(CRivAtom name, int relOp, CRivAtom value)</code>
Description	Construct a <code>CRivQueryAtom</code> where the left- and right-hand sides are the specified <code>CRivAtom</code> objects, and the operator is set to the given value.
Parameters	<p><code>name</code>—the left-hand name part of the query.</p> <p><code>relOp</code>—one of the operator constants defined in <code>IRivOper</code> (<code>==</code>, <code>!=</code>, <code><</code>, <code><=</code>, <code>></code>, <code>>=</code>, <code>In</code> and <code>Not In</code>).</p> <p><code>value</code>—the right-hand value part of the query.</p>
See Also	<code>CRivAtom</code>

3)

Constructor	<code>public CRivQueryAtom(CRivExpr lhExpr, int relOp, CRivExpr rhExpr)</code>
Description	<p>Construct a <code>CRivQueryAtom</code> where the left- and right-hand sides are the specified <code>CRivExpr</code> objects, and the operator is set to the given value.</p> <p>It should be noted that in this release only queries built from <code>CRivExpr</code> objects of type <code>CRivExpr.SUPPLIED</code> will be correctly evaluated. Other types of <code>CRivExpr</code> will be converted to <code>CRivAtom</code> objects of type <code>IRivDataType.RDT_STRING</code>.</p>
Parameters	<p><code>lhExpr</code>—the left-hand expression part of the query.</p> <p><code>relOp</code>—one of the operator constants defined in <code>IRivOper</code> (<code>==</code>, <code>!=</code>, <code><</code>, <code><=</code>, <code>></code>, <code>>=</code>, <code>In</code> and <code>Not In</code>).</p> <p><code>rhExpr</code>—the right-hand expression part of the query.</p>
See Also	<code>CRivExpr</code>

4)

Constructor	<code>public CRivQueryAtom(CRivQueryAtom qryAtmToCopy)</code>
Description	Construct a CRivQueryAtom by copying the supplied CRivQueryAtom.
Parameters	qryAtmToCopy—the query atom to copy.

Method detail

1)

Method	<code>public CRivAtom getRQALeft()</code>
Description	Return the CRivAtom which forms the left-hand side of this query atom.

2)

Method	<code>public int getRQAOperator()</code>
Description	Return the relative operator of the query atom.
Returns	One of the operator constants defined in IRivOper.

3)

Method	<code>public CRivAtom getRQARight()</code>
Description	Return the CRivAtom which forms the right-hand side of this query atom.

4)

Method	<code>public int getRQAValueType()</code>
Description	Return the data type of the CRivAtom which forms the right-hand value side of this CRivQueryAtom.
Returns	One of the data type constants defined in IRivDataType.

5)

Method	<code>public boolean rqaEvalQuery(CRivRecord record)</code>
Description	Evaluate whether the specified <code>CRivRecord</code> satisfies the query represented by this <code>CRivQueryAtom</code> .
Parameters	<code>record</code> —the record to be evaluated to see if it passes the query.
Returns	<code>true</code> , if the record satisfies the query, <code>false</code> otherwise.
Throws	<code>java.lang.NullPointerException</code> —if the specified <code>CRivRecord</code> is null.

6)

Method	<code>public void setRQALeft(CRivAtom name)</code>
Description	Set the left-hand side name part of this query atom to be the given <code>CRivAtom</code> .
Parameters	<code>name</code> —the new <code>CRivAtom</code> name for this query atom.

7)

Method	<code>public void setRQAOperator(int oper)</code>
Description	Set the operator part of this query atom to be the given value.
Parameters	<code>oper</code> —one of the operator constants defined in <code>IRivOper</code> (<code>==</code> , <code>!=</code> , <code><</code> , <code><=</code> , <code>></code> , <code>>=</code> , <code>In</code> and <code>Not In</code>). It should be noted that in this release the operators <code>Matches</code> , <code>Not Matches</code> , <code>Exists</code> , <code>Between</code> , <code>Not Between</code> , <code>Is null</code> , and <code>Is Not null</code> are not supported.

8)

Method	<code>public void setRQARight(CRivAtom value)</code>
Description	Set the right-hand side value part of this query atom to be the given <code>CRivAtom</code> .
Parameters	<code>value</code> —the new <code>CRivAtom</code> value for this query atom.

9)

Method `public String toString()`

Description Return a `String` representation of this `CRivQueryAtom`.

Overrides `toString` in class `java.lang.Object`.

Class CRivRecord

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Figure 4-19 Hierarchy of class CRivRecord



```
public class CRivRecord
extends java.lang.Object
```

Description

This class is the fundamental stored item in all Cisco data engines. It consists of a list of `CRivVarBind` objects, where each `CRivVarBind` is a name=value pair of `CRivAtom` objects. The `CRivVarBinds` in a `CRivRecord` object are also hashed on name for ready retrieval of components. Since the data held in a `CRivAtom` may be a list of other `CRivAtom` objects, or a list of `CRivVarBind` objects, compound objects can be easily maintained by a `CRivRecord`.

An example `CRivRecord`, when printed using the `toString()` method of this class, takes the form below, which is taken from the `master.entityByName` database table in `riv_model`.

```
{
    ObjectID=3;
    EntityName='192.168.1.99';
    Address=['', '', '192.168.1.99', ''];
    Description='Internet Operating System Software';
    EntityType=1;
    EntityOID='1.3.6.1.4.1.9.1.278';
    ActionType=0;
    IsActive=1;
    CreateTime=983892465;
    ChangeTime=983892465;
    ClassName='MainNode';
}
```


See Also

CRivAtom
CRivVarBind

Constructor summary

Constructor	Reference Page
1. <code>public CRivRecord()</code>	116
2. <code>public CRivRecord(CRivRecord recToCopy)</code>	116
3. <code>public CRivRecord(CRivVarBindList vbList)</code>	116

Method summary

Method	Reference Page
1. <code>public boolean equals(Object obj)</code>	117
2. <code>public java.util.Enumeration getRRElements()</code>	117
3. <code>public int getRRSize()</code>	117
4. <code>public CRivAtom getRRValueOf(CRivAtom nameAtom)</code>	117
5. <code>public CRivAtom getRRValueOf(String name)</code>	118
6. <code>public CRivVarBind getRRValuePairOf(CRivAtom nameAtom)</code>	118
7. <code>public CRivVarBind getRRValuePairOf(String name)</code>	118
8. <code>public int rrAddValue(CRivAtom nm, CRivAtom val)</code>	119
9. <code>public int rrAddValue(CRivVarBind vb)</code>	119
10. <code>public int rrDecode(byte[] pktBytes)</code>	119
11. <code>public byte[] rrEncode()</code>	120
12. <code>public int rrInitFromVarList(CRivVarBindList vbList)</code>	120
13. <code>public String rrPrintVal(CRivAtom val, String pad)</code>	120
14. <code>public CRivVarBind rrRemoveValue(CRivAtom nameAtom)</code>	121
15. <code>public CRivVarBind rrRemoveValue(String name)</code>	121

Method	Reference Page
16. <code>public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value)</code>	121
17. <code>public CRivVarBind setRRValueOf(String name, CRivAtom value)</code>	122
18. <code>public String toString()</code>	122

Constructor detail

1)

Constructor `public CRivRecord()`

Description Construct a new, empty `CRivRecord`.

2)

Constructor `public CRivRecord(CRivRecord recToCopy)`

Description Copy a `CRivRecord`. A deep copy is made of all the `CRivVarBinds` in the supplied `CRivRecord`.

Parameters `recToCopy`—the record to be deep copied.

3)

Constructor `public CRivRecord(CRivVarBindList vbList)`

Description Make a `CRivRecord` from a supplied list of `CRivVarBinds`. Note that deep copies are not made up of the `CRivVarBinds` in the supplied list.

Parameters `vbList`—the list of varbinds from which to make the `CRivRecord`.

Method detail

1)

Method `public boolean equals(Object obj)`

Description Compare this record with another for equality.

Parameters `obj`—the object to compare with.

Overrides `equals` in class `Object`.

Returns `true`, if the supplied object is a non-null `CRivRecord`, containing the same number of `CRivVarBinds`, and the same name/value pairs. `false`, otherwise.

2)

Method `public java.util.Enumeration getRRElements()`

Description Return an enumeration of the component `CRivVarBinds` of this record. The returned `Enumeration` object will generate all items in this record. The first item generated is the item at index 0, then the item at index 1, and so on.

3)

Method `public int getRRSize()`

Description Return the number of `CRivVarBinds` in this record.

4)

Method `public CRivAtom getRRValueOf(CRivAtom nameAtom)`

Description Get the value of a name field. Returns `null` if the field does not appear in this record.

Parameters `nameAtom`—the name of the field to obtain the value of.

Returns The value of the specified name field as a `CRivAtom`.

5)

Method `public CRivAtom getRRValueOf(String name)`

Description Get the value of a name field. Returns `null` if the field does not appear in this record.

Parameters `name`—the name of the field to obtain the value of.

Returns The value of the specified name field as a `CRivAtom`.

6)

Method `public CRivVarBind getRRValuePairOf(CRivAtom nameAtom)`

Description Get the `CRivVarBind` for a given name field. Returns `null` if the field does not appear in this record.

Parameters `nameAtom`—the name of the `CRivVarBind` to obtain.

Returns The `CRivVarBind` for the given name field.

7)

Method `public CRivVarBind getRRValuePairOf(String name)`

Description Get the `CRivVarBind` for a given name field. Returns `null` if the field does not appear in this record.

Parameters `name`—the name of the `CRivVarBind` to obtain.

Returns The `CRivVarBind` for the given name field.

8)

Method `public int rrAddValue(CRivAtom nm, CRivAtom val)`

Description Add a new name=value pair into the CRivRecord.

Parameters nm—the name part of the CRivVarBind to add to the record.
val—the value part of the CRivVarBind to add to the record.

Returns An int, from CRivException, signalling whether the CRivVarBind was added to the record, e.g., CRivException.RIV_OK if successfully added.

See Also `rrAddValue(CRivVarBind)`

9)

Method `public int rrAddValue(CRivVarBind vb)`

Description Add the CRivVarBind to the record. If the record already contains an entry with the same name as the specified CRivVarBind the method will return CRivException.DUPLICATE_KEY_INSERT and the new value will not get added to the record.

Parameters vb—the CRivVarBind to add to the record.

Returns An int, from CRivException, signalling whether the CRivVarBind was added to the record, e.g., CRivException.RIV_OK if successfully added.

See Also `rrAddValue(CRivAtom nm, CRivAtom val)`

10)

Method `public int rrDecode(byte[] pktBytes)`

Description Set the values from a supplied ROMP packet.

Parameters pktBytes—the ROMP packet to be decoded into this CRivRecord.

Returns An int signifying whether the byte[] was successfully decoded into a CRivRecord, one of the constants defined by CRivException, e.g., CRivException.RIV_OK if successful.

11)

Method `public byte[] rrEncode()`

Description Return the values held in the record as a ROMP packet.

Returns The encoded ROMP packet obtained via `CRivROMPer` as a `byte []`.

Throws `CRivException`—if an error occurred while encoding the values held in this `CRivRecord`.

See Also `CRivROMPer`

12)

Method `public int rrInitFromVarList(CRivVarBindList vbList)`

Description Initialize from a list of `CRivVarBinds`.

Note This does not copy memory; the supplied list of `CRivVarBinds` is simply added into the list held in this record. This method does not clear out internal tables, so the record increases by the supplied list.

If a `CRivVarBind` in the supplied list has the same name as a `CRivVarBind` in this record, it will not be added in.

Parameters `vbList`—the list of `CRivVarBinds` from which this `CRivRecord` is initialised.

Returns An `int`, one of the constants defined in `CRivException`, signifying whether the record was successfully initialised e.g., `CRivException.RIV_OK` if successful.

13)

Method `public String rrPrintVal(CRivAtom val, String pad)`

Description Print the given `CRivAtom` in a readable format.

Parameters `val`—the value to be printed.
`pad`—padding to add between fields when printing a `CRivAtom.RDT_OBJECT`.

Returns A `String` representation of the `CRivAtom`.

14)

Method `public CRivVarBind rrRemoveValue(CRivAtom nameAtom)`

Description Remove the `CRivVarBind` with the given `CRivAtom` name from the record.

Parameters `nameAtom`—the name of the `CRivVarBind` to be removed.

Returns The `CRivVarBind` which was removed from the `CRivRecord`, or null if the record did not contain a `CRivVarBind` with the given name.

15)

Method `public CRivVarBind rrRemoveValue(String name)`

Description Remove the `CRivVarBind` with the given name from the record.

Parameters `name`—the name of the `CRivVarBind` to be removed.

Returns The `CRivVarBind` which was removed from the `CRivRecord`, or null if the record did not contain a `CRivVarBind` with the given name.

16)

Method `public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value)`

Description Set the value of the `CRivVarBind` with the given name to the specified value.

Parameters `name`—the name of the `CRivVarBind` whose value will be set.
`value`—the value of the `CRivVarBind` to be set.

Returns The new `CRivVarBind` with the specified value, or null if the record did not contain a `CRivVarBind` with the given name, in which case the new `CRivVarBind` is not added to the table (use `rrAddValue()` instead).

17)

Method `public CRivVarBind setRRValueOf(String name, CRivAtom value)`

Description Set the value of the `CRivVarBind` with the given name to the specified value.

Parameters `name`—the name of the `CRivVarBind` whose value will be set.
`value`—the value of the `CRivVarBind` to be set.

Returns The new `CRivVarBind` with the specified value, or null if the record did not contain a `CRivVarBind` with the given name, in which case the new `CRivVarBind` is not added to the table (use `rrAddValue()` instead).

18)

Method `public synchronized String toString()`

Description Returns a `String` representation of this record, containing the `String` representation of each element.

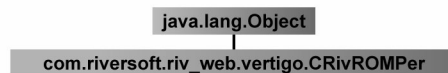
Overrides `toString` in class `java.lang.Object`.

Class CRivROMPer

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-20 Hierarchy of class *CRivROMPer*



```
public class CRivROMPer
extends java.lang.Object
```


Description

RiverSoft Object Multicast Protocol (ROMP) is an internal network serialization protocol used to distribute executables. RiverSoft decided to implement ROMP because it improves efficiency, as ROMP is fast and extensible. ROMP transmits objects between processes by converting complex objects to streams of data. ROMP intends to serialize anything, thus making it serial (linear from a to b to c) and not parallel. This means that a ROMP transmission of information has to wait for another to finish before it can start, or has to wait for everything else to finish before it can commence.

This class uses low level encode/decode methods for ROMP format packets.

CRivROMPer

Application-level code should rarely need to call this class directly. It is used within CRivRecord to encode, or decode, network data.

See Also

rrEncode, rrDecode (CRivRecord)

Field summary

Field	Reference Page
1. public static final int RIV_ROMP_ASN1	125
2. public static final int RIV_ROMP_BYTE_ARRAY	125
3. public static final int RIV_ROMP_FLOAT	125
4. public static final int RIV_ROMP_INT	125
5. public static final int RIV_ROMP_LIST_TYPE	126
6. public static final int RIV_ROMP_LONG	126
7. public static final int RIV_ROMP_LONGLONG	126
8. public static final int RIV_ROMP_OBJECT_TYPE	126
9. public static final int RIV_ROMP_STRING	126
10. public static final int RIV_ROMP_UNDEFINED	127

Constructor summary

Constructor	Reference Page
1. <code>public CRivROMPer()</code>	127
2. <code>public CRivROMPer(byte[] bytes, boolean isBigEndian)</code>	127

Method summary

Method	Reference Page
1. <code>public CRivVarBind <i>getRRNextAttrVal</i>()</code>	127
2. <code>public byte[] <i>getRRPacket</i>()</code>	127
3. <code>public int <i>getRRPacketLength</i>()</code>	128
4. <code>public int <i>rrAddAttrVal</i>(CRivVarBind varbind)</code>	128
5. <code>public int <i>rrAppendFrom</i>(CRivROMPer romper)</code>	128
6. <code>protected CRivAtom <i>rrDecodeASN1</i>(byte[] pkt, int offset, int valueLength)</code>	129
7. <code>protected CRivAtom <i>rrDecodeByteArray</i>(byte[] pkt, int offset, int valueLength)</code>	129
8. <code>protected CRivAtom <i>rrDecodeFloat</i>(byte[] pkt, int offset, int valueLength)</code>	129
9. <code>protected CRivAtom <i>rrDecodeInt</i>(byte[] pkt, int offset, int valueLength)</code>	130
10. <code>protected CRivAtom <i>rrDecodeLong</i>(byte[] pkt, int offset, int valueLength)</code>	130
11. <code>protected String <i>rrDecodeName</i>(byte[] pkt, int offset)</code>	130
12. <code>protected CRivAtom <i>rrDecodeString</i>(byte[] pkt, int offset, int valueLength)</code>	131
13. <code>protected int <i>rrDecodeType</i>(byte[] pkt, int offset)</code>	131
14. <code>protected int <i>rrDecodeValueLength</i>(byte[] pkt, int offset)</code>	131
15. <code>protected void <i>rrEncodeASN1</i>(String asn1Str)</code>	132
16. <code>protected void <i>rrEncodeByteArray</i>(byte[] byteArr)</code>	132
17. <code>protected void <i>rrEncodeFloat</i>(float val)</code>	133
18. <code>protected void <i>rrEncodeHeader</i>(int type, String fieldName, int valLen)</code>	133

Method	Reference Page
19. protected void <i>rrEncodeInt</i> (int val)	133
20. protected void <i>rrEncodeLong</i> (long val)	134
21. protected void <i>rrEncodeString</i> (String str)	134
22. public void <i>rrUpPacketSize</i> (int increase)	134

Field detail

1)

Field `public static final int RIV_ROMP_ASN1`

Description Object ID (asn1) data type.

Value 3

2)

Field `public static final int RIV_ROMP_BYTE_ARRAY`

Description Byte array data type.

Value 7

3)

Field `public static final int RIV_ROMP_FLOAT`

Description Float data type.

Value 4

4)

Field `public static final int RIV_ROMP_INT`

Description Integer data type.

Value 1

5)

Field	<code>public static final int RIV_ROMP_LIST_TYPE</code>
Description	List data type. Data of this type will ultimately be decoded/encoded as a <code>CRivAtom</code> which is a list of <code>CRivAtoms</code> .
Value	64

6)

Field	<code>public static final int RIV_ROMP_LONG</code>
Description	Long data type.
Value	5

7)

Field	<code>public static final int RIV_ROMP_LOGLONG</code>
Description	Data type corresponding to the C++ unsigned long (i.e., 64 bit) data type. In Java, this will be decoded/encoded as a <code>RIV_ROMP_LONG</code> type.
Value	6

8)

Field	<code>public static final int RIV_ROMP_OBJECT_TYPE</code>
Description	Object data type. Data of this type will ultimately be decoded/encoded as a <code>CRivAtom</code> which is a list of <code>CRivAtoms</code> .
Value	128

9)

Field	<code>public static final int RIV_ROMP_STRING</code>
Description	String data type.
Value	2

10)

Field	<code>public static final int RIV_ROMP_UNDEFINED</code>
Description	Undefined data type.
Value	8

Constructor detail**1)**

Constructor	<code>public CRivROMPer()</code>
Description	Construct an empty <code>CRivROMPer</code> with no data. Any data added to the object is assumed to be in network byte order.

2)

Constructor	<code>public CRivROMPer(byte[] bytes, boolean isBigEndian)</code>
Description	Construct a <code>CRivROMPer</code> object with the specified data as a byte array. The data may be processed according to whether it is in network or host byte order.
Parameters	<code>bytes</code> —the data to be processed by this object. <code>isBigEndian</code> —if <code>true</code> , the data will be processed in network byte order. If <code>false</code> , it is processed in host byte order.

Method detail**1)**

Method	<code>public CRivVarBind getRRNextAttrVal()</code>
Description	Get the next attribute value.
Returns	The next <code>CRivVarBind</code> in the packet.

2)

Method	<code>public byte[] getRRPacket()</code>
Description	Return the packet data in this <code>CRivROMPer</code> .

3)

Method `public int getRRPacketLength()`

Description Return the length of the ROMP packet.

Returns The number of bytes in the packet, or zero if the packet has not been initialized.

4)

Method `public int rrAddAttrVal(CRivVarBind varbind)`

Description Encode a CRivVarBind and add it to the ROMP packet.

Parameters `varbind`—the CRivVarBind to be decoded and added to this packet.

Returns An int, one of the values from CRivException, indicating whether the CRivVarBind was added successfully to the CRivROMPer packet; e.g., CRivException.RIV_OK if successful.

Throws CRivException—if an unexpected error occurs encoding the CRivVarBind.

5)

Method `public int rrAppendFrom(CRivROMPer romper)`

Description Append data held in another CRivROMPer packet onto the end of this ROMP packet. If there is not enough space to add on this additional packet the size of this ROMP packet will automatically be increased to allow for the additional data.

Parameters `romper`—the ROMP packet to be appended to ‘this’ one.

Returns An int, one of the constants defined in CRivException indicating whether the CRivROMPer was successfully appended; e.g., CRivException.RIV_OK if successful.

6)

Method	<code>protected CRivAtom rrDecodeASN1(byte[] pkt, int offset, int valueLength)</code>
Description	Get a <code>CRivAtom</code> of type <code>IRivDataType.RDT_ASN1</code> from the next <code>valueLength</code> bytes.
Parameters	<code>pkt</code> —the packet containing the value to be decoded. <code>offset</code> —the position in the packet from which to start decoding. <code>valueLength</code> —the number of bytes to decode.
Returns	The <code>CRivAtom</code> of type <code>IRivDataType.RDT_ASN1</code> which has been decoded from the specified section of the packet.

7)

Method	<code>protected CRivAtom rrDecodeByteArray(byte[] pkt, int offset, int valueLength)</code>
Description	Get a <code>CRivAtom</code> of type <code>IRivDataType.RDT_BYTE_ARRAY</code> from the next <code>valueLength</code> bytes.
Parameters	<code>pkt</code> —the packet containing the value to be decoded. <code>offset</code> —the position in the packet from which to start decoding. <code>valueLength</code> —the number of bytes to decode.
Returns	The <code>CRivAtom</code> of type <code>IRivDataType.RDT_BYTE_ARRAY</code> which has been decoded from the specified section of the packet.

8)

Method	<code>protected CRivAtom rrDecodeFloat(byte[] pkt, int offset, int valueLength)</code>
Description	Get a <code>CRivAtom</code> of type <code>IRivDataType.RDT_FLOAT</code> from the next four bytes.
Parameters	<code>pkt</code> —the packet containing the value to be decoded. <code>offset</code> —the position in the packet from which to start decoding. <code>valueLength</code> —the number of bytes to advance the cursor after decoding.
Returns	The <code>CRivAtom</code> of type <code>IRivDataType.RDT_FLOAT</code> which has been decoded from the specified section of the packet.

9)

Method	<code>protected CRivAtom rrDecodeInt(byte[] pkt, int offset, int valueLength)</code>
Description	Get a <code>CRivAtom</code> of type <code>IRivDataType.RDT_INT</code> from the next four bytes.
Parameters	<code>pkt</code> —the packet containing the value to be decoded. <code>offset</code> —the position in the packet from which to start decoding. <code>valueLength</code> —the number of bytes to advance the cursor after decoding.
Returns	The <code>CRivAtom</code> of type <code>IRivDataType.RDT_INT</code> which has been decoded from the specified section of the packet.

10)

Method	<code>protected CRivAtom rrDecodeLong(byte[] pkt, int offset, int valueLength)</code>
Description	Get a <code>CRivAtom</code> of type <code>IRivDataType.RDT_LONG</code> from the next eight bytes.
Parameters	<code>pkt</code> —the packet containing the value to be decoded. <code>offset</code> —the position in the packet from which to start decoding. <code>valueLength</code> —the number of bytes to advance the cursor after decoding.
Returns	The <code>CRivAtom</code> of type <code>IRivDataType.RDT_LONG</code> which has been decoded from the specified section of the packet.

11)

Method	<code>protected String rrDecodeName(byte[] pkt, int offset)</code>
Description	Return the name part of a <code>CRivVarBind</code> as a <code>String</code> .
Parameters	<code>pkt</code> —the packet containing the value to be decoded. <code>offset</code> —the position in the packet from which to start decoding.

12)

Method `protected CRivAtom rrDecodeString(byte[] pkt, int offset, int valueLength)`

Description Get a `CRivAtom` of type `IRivDataType.RDT_STRING` from the next `valueLength` bytes.

Parameters `pkt`—the packet containing the value to be decoded.
`offset`—the position in the packet from which to start decoding.
`valueLength`—the number of bytes to decode.

Returns The `CRivAtom` of type `IRivDataType.RDT_STRING` which has been decoded from the specified section of the packet.

13)

Method `protected int rrDecodeType(byte[] pkt, int offset)`

Description Decode the type of the value part of a `CRivVarBind` as an `int`.

Parameters `pkt`—the packet containing the value to be decoded.
`offset`—the position in the packet from which to start decoding.

Returns The decoded type of the value part of a `CRivVarBind` as an `int`.

14)

Method `protected int rrDecodeValueLength(byte[] pkt, int offset)`

Description Decode the length of the value part of a `CRivVarBind` as an `int`.

Parameters `pkt`—the packet containing the value to be decoded.
`offset`—the position in the packet from which to start decoding.

Returns The decoded length of the value part of a `CRivVarBind` as an `int`.

15)

Method `protected void rrEncodeASN1(String asn1Str)`

Description Encode a `String` representation of a `CRivASN1Address`, by converting it into a `byte[]` and adding it onto the packet. The packet must already contain enough bytes to add on the `String` as bytes. If not, `rrUpPacketSize(int)` should be called first.

Parameters `asn1Str`—the `String` representation of a `CRivASN1Address` to encode.

Throws `ArrayIndexOutOfBoundsException`—if the packet did not contain enough spare bytes to add on the `int`.
`NullPointerException`—if the argument `String` is `null`.

See Also `rrUpPacketSize(int)`, `CRivASN1Address`

16)

Method `protected void rrEncodeByteArray(byte[] byteArr)`

Description Encode a `byte[]` by simply adding it onto the packet. The packet must already contain enough bytes to add on the specified array. If not `rrUpPacketSize(int)` should be called first.

Parameters `byteArr`—the `byte` array to be encoded.

Throws `ArrayIndexOutOfBoundsException`—if the packet did not contain enough spare bytes to add on the array.
`NullPointerException`—if the argument is `null`.

See Also `rrUpPacketSize(int)`

17)

Method `protected void rrEncodeFloat(float val)`

Description Encode a float by converting it into an array of 4 bytes and adding it onto the packet. The packet must already contain enough bytes to add on the float as bytes. If not, `rrUpPacketSize(int)` should be called first.

Parameters `val`—the float to be encoded.

Throws `ArrayIndexOutOfBoundsException`—if the packet did not contain enough spare bytes to add on the float.

See Also `rrUpPacketSize(int)`

18)

Method `protected void rrEncodeHeader(int type, String fieldName, int valLen)`

Description Encode the specified data type, field name and value length fields.

Parameters `type`—the type of the data to be encoded.

`fieldName`—the field name of the data to be encoded.

`valLen`—the length of the value to be encoded.

19)

Method `protected void rrEncodeInt(int val)`

Description Encode an int by converting it into an array of 4 bytes and adding it onto the packet. The packet must already contain enough bytes to add on the int as bytes. If not, `rrUpPacketSize(int)` should be called first.

Parameters `val`—the int to be encoded.

Throws `ArrayIndexOutOfBoundsException`—if the packet did not contain enough spare bytes to add on the int.

See Also `rrUpPacketSize(int)`

20)

Method `protected void rrEncodeLong(long val)`

Description Encode a `long` by converting it into an array of 8 bytes and adding it onto the packet. The packet must already contain enough bytes to add on the `long` as bytes. If not, `rrUpPacketSize(int)` should be called first.

Parameters `val`—the `long` to be encoded.

Throws `ArrayIndexOutOfBoundsException`—if the packet did not contain enough spare bytes to add on the `long`.

See Also `rrUpPacketSize(int)`

21)

Method `protected void rrEncodeString(String str)`

Description Encode a `String`, by converting it into a `byte []` and adding it onto the packet. The packet must already contain enough bytes to add on the `String` as bytes. If not, `rrUpPacketSize(int)` should be called first.

Parameters `val`—the `String` to be encoded.

Throws `ArrayIndexOutOfBoundsException`—if the packet did not contain enough spare bytes to add on the `String`.

`NullPointerException`—if the argument `String` is `null`.

See Also `rrUpPacketSize(int)`

22)

Method `public void rrUpPacketSize(int increase)`

Description Increase the size of the ROMP packet by the specified size. If the packet is currently `null`, it initializes the packet with a size equal to the specified increase.

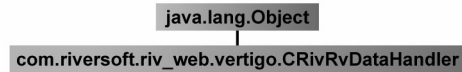
Parameters `increase`—the amount that the size of the ROMP packet should be increased by.

Class CRivRvDataHandler

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-21 Hierarchy of class CRivRvDataHandler



```

public class CRivRvDataHandler
  extends java.lang.Object
  implements COM.TIBCO.rv.RvDataCallback, COM.TIBCO.rv.RvTimeOutCallback,
  java.lang.Runnable
  
```

Description

This handler class is used to receive and process data for a Rendezvous listener. It implements the Rendezvous callback interfaces `RvDataCallback`, and `RvTimeOutCallback`. It also implements `java.lang.Runnable` to process the data into arrays of `CRivRecords` in a separate thread. These `CRivRecords` are then passed onto the object implementing the `IRivRecordListener` interface which was passed in when this object was created. A single `CRivRvDataHandler` object should only be used to handle a single request from the `CRivClient` at a time.

See Also

[CRivClient](#)
[CRivRecord](#)
[IRivRecordListener](#)

Field summary

Field	Reference Page
1. public static final int <i>RRDH_DATA_RECEIVED</i>	136
2. public static final int <i>RRDH_TIMED_OUT</i>	137
3. public static final int <i>RRDH_UNKNOWN</i>	137

Constructor summary

Constructor	Reference Page
1. <code>public CRivRvDataHandler(IRivRecordListener recListener)</code>	137

Method summary

Method	Reference Page
1. <code>protected java.lang.Thread <i>getRRDHProcessorThread</i>()</code>	137
2. <code>public int <i>getRRDHStatus</i>()</code>	138
3. <code>public void <i>onData</i>(String subject, COM.TIBCO.rv.RvSender replySender, Object data, COM.TIBCO.rv.RvListener invoker)</code>	138
4. <code>public void <i>onTimeout</i>(COM.TIBCO.rv.RvListener invoker)</code>	138
5. <code>public void <i>rrdhCancelListener</i>()</code>	139
6. <code>public CRivRecord[] <i>rrdhConvertAtoms</i>(CRivAtom[] byteArrayAtoms)</code>	139
7. <code>public void <i>rrdhProcessData</i>()</code>	139
8. <code>public void <i>rrdhStartThread</i>()</code>	140
9. <code>public void <i>rrdhStopThread</i>()</code>	140
10. <code>public void <i>rrdhWait</i>()</code>	140
11. <code>public void <i>run</i>()</code>	140
12. <code>public void <i>setRRDHStatus</i>(int status)</code>	140

Field detail

1)

Field	<code>public static final int RRDH_DATA_RECEIVED</code>
Description	Status constant indicating that all the data has been received in the transport layer.
Value	0

2)

Field	<code>public static final int RRDH_TIMED_OUT</code>
Description	Status constant indicating that the Rendezvous listener has timed out before any data has been received from the chosen domain.
Value	1

3)

Field	<code>public static final int RRDH_UNKNOWN</code>
Description	Status of callback unknown, i.e., no data, or no time out message, has been received.
Value	2

Constructor detail

1)

Constructor	<code>public CRivRvDataHandler(IRivRecordListener recListener)</code>
Description	Construct a CRivRvDataHandler.
Parameters	<code>recListener</code> —the IRivRecordListener to which any CRivRecords processed from the transport layer should be passed onto. This can be null.
See Also	IRivRecordListener

Method detail

1)

Method	<code>protected java.lang.Thread getRRDHProcessorThread()</code>
Description	Return the thread that processes the data received from Rendezvous. Users of this class should not need to call this method.

2)

Method	<code>public int getRRDHStatus()</code>
Description	Return the current status of the Rendezvous data download.
Returns	One of the constants defined in this class which indicates whether the status is unknown or if data has been provided.

3)

Method	<code>public void onData(String subject, COM.TIBCO.rv.RvSender replySender, Object data, COM.TIBCO.rv.RvListener invoker)</code>
Description	The callback method from the RvDataCallback interface. The CRivClient which appointed this handler is stored in the invoker argument. Subclasses should override this method to provide any additional functionality, if required.
Parameters	<p><code>subject</code>—the new sender sends messages addressed to this subject name, which may denote either a broadcast subject or an inbox.</p> <p><code>replySender</code>—if the arriving message carries a reply subject, this parameter receives a newly created sender activity that sends to that reply subject. Otherwise, this parameter receives null.</p> <p><code>data</code>—this parameter receives the data object.</p> <p><code>invoker</code>—each timer activity appoints a handler object to process timer events on its behalf. This parameter receives the timer activity that appointed this handler.</p>
Specified By	<code>onData</code> in interface <code>COM.TIBCO.rv.RvDataCallback</code> .

4)

Method	<code>public void onTimeOut(COM.TIBCO.rv.RvListener invoker)</code>
Description	The callback method from the RvTimeOutCallback interface. The CRivClient which appointed this handler is stored in the invoker argument. It sets the status to <code>RDC_TIMED_OUT</code> and cancels listening in the Rendezvous subject. The IRivRecordListener for this handler will be notified of the timeout through its <code>rrlTimeOutReceived()</code> method. Subclasses should override this method to provide any additional functionality.

Parameters `invoker`—each timer activity appoints a handler object to process timer events on its behalf. This parameter receives the timer activity that appointed this handler.

Specified By `onTimeOut` in interface `COM.TIBCO.rv.RvTimeOutCallback`

5)

Method `public void rrdhCancelListener()`

Description Cancel Rendezvous listening interest in the subject. It is recommended that the object that created this handler calls this method when it has finished listening to the Rendezvous subject of interest.

6)

Method `public CRivRecord[] rrdhConvertAtoms(CRivAtom[] byteArrayAtoms)`

Description Convenience method to convert an array of `CRivAtom` objects of type `IRivDataType.RDT_BYTE_ARRAY` into an array of `CRivRecords`.

Parameters `byteArrayAtoms`—the array of `CRivAtom` objects, which are of type `IRivDataType.RDT_BYTE_ARRAY`, to convert.

Returns An array of `CRivRecords` which have been decoded from a specified array of `CRivAtoms`.

7)

Method `public void rrdhProcessData()`

Description Process the decoded ROMP packets stored in the transport layer object into `CRivRecords`. This method is called from within the `run()` method. Users of this class should not need to call this method directly.

8)

Method `public void rrdhStartThread()`

Description Start the thread that processes the incoming data from Rendezvous.

9)

Method `public void rrdhStopThread()`

Description Stop the data processing thread. It is recommended that this method is called as soon as the callback has finished its work, e.g., it has received all records from a query.

10)

Method `public void rrdhWait()`

Description Loop round the data processing thread until either all the data has been received, or `onTimeout()` has been called.

11)

Method `public void run()`

Description Implemented run method from the `Runnable` interface. Uses the `CRivTransport` transport layer object to process the Rendezvous messages which have arrived.

Specified `run` in interface `java.lang.Runnable`
By

12)

Method `public void setRRDHStatus(int status)`

Description Set the status of the Rendezvous download.

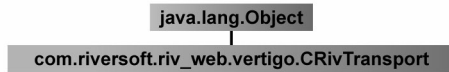
Parameters `status`—one of the status constants defined in this class.

Class CRivTransport

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-22 Hierarchy of class CRivTransport



```
public class CRivTransport
extends java.lang.Object
```

Description

A transport layer object used by every sender/receiver to process data to network, or network to data.

Field summary

Field	Reference Page
1. public static final int <i>RT_CLM_NOT_THERE</i>	143
2. public static final int <i>RT_DB_NOT_THERE</i>	143
3. public static final int <i>RT_INCOMPLETE_ROW</i>	143
4. public static final int <i>RT_INCORRECT_TYPE</i>	143
5. public static final int <i>RT_NO_ERROR</i>	143
6. public static final int <i>RT_PERM_DENIED</i>	144
7. public static final int <i>RT_SEARCH_FAILED</i>	144
8. public static final int <i>RT_TBL_NOT_THERE</i>	144
9. public static final int <i>SENT_PACKET_HEADER_LENGTH</i>	144

Constructor summary

Constructor	Reference Page
1. public CRivTransport()	145
2. public CRivTransport (boolean isBigEndian)	145

Method summary

Method	Reference Page
1. public static byte[] getRTBytesForFloat(float floatVal, boolean bigEndian)	145
2. public static byte[] getRTBytesForInt (int intVal, boolean bigEndian)	145
3. public static byte[] getRTBytesForLong (long longVal, boolean bigEndian)	146
4. public java.util.Vector getRTDecodes ()	146
5. public int getRTError ()	146
6. public static float getRTFloatFromBytes (byte[] bytes, int offset, boolean bigEndian)	146
7. public static int getRTIntFromBytes (byte[] bytes, int offset, boolean bigEndian)	147
8. public static long getRTLLongFromBytes (byte[] bytes, int offset, boolean bigEndian)	147
9. public boolean isRTDone ()	147
10. public void rtAddMessage (COM.TIBCO.rv.RvMsg msg)	147
11. public void rtDecodePktHeader (byte[] packet)	148
12. public void rtPrintError (String userText)	148
13. public void rtPurge ()	148
14. public static float rtReadFloat (byte[] bytes)	148
15. public static int rtReadInt (byte[] bytes)	148
16. public static long rtReadLong (byte[] bytes)	149
17. public static byte[] rtSwapByteArray (byte[] byteArray)	149
18. public void setRTError (int error)	149

Field detail

1)

Field	<code>public static final int RT_CLM_NOT_THERE</code>
Description	Error code indicating that the specified column does not exist.
Value	4

2)

Field	<code>public static final int RT_DB_NOT_THERE</code>
Description	Error code indicating that the desired database does not exist.
Value	2

3)

Field	<code>public static final int RT_INCOMPLETE_ROW</code>
Description	Error code indicating an attempt to insert an incomplete row.
Value	6

4)

Field	<code>public static final int RT_INCORRECT_TYPE</code>
Description	Error code indicating that an assignment between incorrect types is not allowed.
Value	5

5)

Field	<code>public static final int RT_NO_ERROR</code>
Description	Code indicating no error has occurred. The transaction is either complete, or is yet to encounter an error.
Value	0

6)

Field	<code>public static final int RT_PERM_DENIED</code>
Description	Error code indicating that permission is denied.
Value	1

7)

Field	<code>public static final int RT_SEARCH_FAILED</code>
Description	Error code indicating that the query has zero matches.
Value	7

8)

Field	<code>public static final int RT_TBL_NOT_THERE</code>
Description	Error code indicating that the desired table does not exist.
Value	3

9)

Field	<code>public static final int SENT_PACKET_HEADER_LENGTH</code>
Description	The length of the sent packet header.
Value	28

Constructor detail

1)

Constructor	<code>public CRivTransport()</code>
Description	Make a client side layer object in which the data is expected to arrive in the transport object as Big Endian, i.e., in network byte order.

2)

Constructor	<code>public CRivTransport(boolean isBigEndian)</code>
Description	Make a client side layer object.
Parameters	<code>isBigEndian</code> —if <code>true</code> , the data is assumed to arrive in network byte order.

Method detail

1)

Method	<code>public static byte[] getRTBytesForFloat(float floatVal, boolean bigEndian)</code>
Description	Convert the <code>float</code> argument to an array of 4 bytes.
Parameters	<code>floatval</code> —the <code>float</code> to be converted. <code>bigEndian</code> —if <code>true</code> , the resulting byte array is in network byte order.
Returns	The specified <code>float</code> as an array of bytes.

2)

Method	<code>public static byte[] getRTBytesForInt(int intVal, boolean bigEndian)</code>
Description	Convert the <code>int</code> argument to an array of 4 bytes.
Parameters	<code>intVal</code> —the <code>int</code> to be converted. <code>bigEndian</code> —if <code>true</code> , the resulting byte array is in network byte order.
Returns	The specified <code>int</code> as an array of bytes.

3)

Method `public static byte[] getRTBytesForLong(long longVal, boolean bigEndian)`

Description Convert the `long` argument to an array of 8 bytes.

Parameters `longVal`—the `int` to be converted.
`bigEndian`—if `true`, the resulting byte array is in network byte order.

Returns The specified `long` as an array of bytes.

4)

Method `public java.util.Vector getRTDecodes()`

Description Get a list of ROMP packets.

Returns The list of ROMP packets held in this transport object as a vector of `CRivAtoms`.

5)

Method `public int getRTError()`

Description Get the current error code.

Returns The current error code, one of the error codes defined by this class.

6)

Method `public static float getRTFloatFromBytes(byte[] bytes, int offset, boolean bigEndian)`

Description Factory method to extract a `float` from an array of bytes.

Parameters `bytes`—the array of bytes from which the `float` will be read.
`offset`—the position in the packet from which to start reading.
`bigEndian`—if `true`, the sequence of bytes will be read in network byte order.

Returns The specified byte array as a `float`.

7)

Method	<code>public static int getRTIntFromBytes(byte[] bytes, int offset, boolean bigEndian)</code>
Description	Factory method to extract an <code>int</code> from an array of <code>bytes</code> .
Parameters	<code>bytes</code> —the array of <code>bytes</code> from which the <code>int</code> will be read. <code>offset</code> —the position in the packet from which to start reading. <code>bigEndian</code> —if <code>true</code> , the sequence of <code>bytes</code> will be read in network byte order.
Returns	The specified <code>byte</code> array as an <code>int</code> .

8)

Method	<code>public static long getRTLLongFromBytes(byte[] bytes, int offset, boolean bigEndian)</code>
Description	Factory method to extract a <code>long</code> from an array of <code>bytes</code> .
Parameters	<code>bytes</code> —the array of <code>bytes</code> from which the <code>float</code> will be read. <code>offset</code> —the position in the packet from which to start reading. <code>bigEndian</code> —if <code>true</code> , the sequence of <code>bytes</code> will be read in network byte order.
Returns	The specified <code>byte</code> array as a <code>long</code> .

9)

Method	<code>public boolean isRTDone()</code>
Description	Determine whether the <code>CRivTransport</code> object has received and processed the last packet in the sequence.
Returns	<code>true</code> , if all the data has been received and processed, <code>false</code> otherwise.

10)

Method	<code>public void rtAddMessage(COM.TIBCO.rv.RvMsg msg)</code>
Description	Receive a packet, process to net ready, and update the done flag.
Parameters	<code>msg</code> —the message to be added and then decoded by this <code>CRivTransport</code> object.

11)

Method `public void rtDecodePktHeader(byte[] packet)`

Description Decode the packet header to strip out the sequence number and the 'last in sequence' number.

Parameters `packet`—the packet containing the header to be decoded.

12)

Method `public void rtPrintError(String userText)`

Description Display the error code as a `String`.

Parameters `userText`—the `String` to be displayed in the error code.

13)

Method `public void rtPurge()`

Description Purge for new data.

14)

Method `public static float rtReadFloat(byte[] bytes)`

Description Factory method to convert an array of bytes to a float.

Parameters `bytes`—array of bytes to be converted to a float.

Returns The first four bytes from the given array interpreted as a float.

Throws `java.io.IOException`—if an I/O error occurs during the reading of the byte array.

15)

Method `public static int rtReadInt(byte[] bytes)`

Description Factory method to convert an array of bytes to an int.

Parameters `bytes`—the array of bytes to be converted to an int.

Returns The first four bytes from the given array interpreted as an `int`.

Throws `java.io.IOException`—if an I/O error occurs during the reading of the `byte` array.

16)

Method `public static long rtReadLong(byte[] bytes)`

Description Factory method to convert an array of bytes to a long.

Parameters `bytes`—the array of bytes to be converted to a long.

Returns The first 8 bytes from the given array interpreted as a long.

Throws `java.io.IOException`—if an I/O error occurs during the reading of the `byte` array.

17)

Method `public static byte[] rtSwapByteArray(byte[] byteArray)`

Description Utility method to swap the order of an array of bytes. Used for Big Endian to Little Endian conversion.

Parameters `byteArray`—the array of bytes whose order will be swapped.

Returns An array which is the reverse of the specified array.

18)

Method `public void setRTError(int error)`

Description Set the current error code.

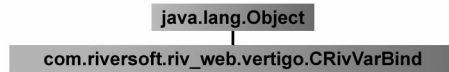
Parameters `error`—the error code.

Class CRivVarBind

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-23 Hierarchy of class *CRivVarBind*



```
public class CRivVarBind
extends java.lang.Object
```

Description

This class is used to store paired values in the form name=value, where both name and value are *CRivAtom* objects. This is used in data stores, such as *CRivRecord*, which hold a record of data in a list of *CRivVarBind* objects.

See Also

CRivAtom
CRivRecord

Constructor summary

Constructor	Reference Page
1. <code>public CRivVarBind(CRivVarBind arg)</code>	151
2. <code>public CRivVarBind(CRivAtom nm, CRivAtom vl)</code>	151
3. <code>public CRivVarBind(String nm, CRivAtom vl)</code>	152
4. <code>public CRivVarBind(String nm, String vl)</code>	152
5. <code>public CRivVarBind(String nm, int vl)</code>	152
6. <code>public CRivVarBind(String nm, long vl)</code>	152

Method summary

Method	Reference Page
1. <code>public boolean <i>equals</i>(Object obj)</code>	153
2. <code>public CRivAtom <i>getRVBName</i>()</code>	153
3. <code>public String <i>getRVBValAsString</i>()</code>	153
4. <code>public CRivAtom <i>getRVBValue</i>()</code>	153
5. <code>public void <i>setRVBName</i>(CRivAtom name)</code>	154
6. <code>public void <i>setRVBValue</i>(CRivAtom value)</code>	154
7. <code>public String <i>toString</i>()</code>	154

Constructor detail

1)

Constructor `public CRivVarBind(CRivVarBind arg)`

Description Copy constructor.

Parameters `arg`—the CRivVarbind to copy.

2)

Constructor `public CRivVarBind(CRivAtom nm, CRivAtom vl)`

Description Make a CRivVarBind of the supplied name and value.

Parameters `nm`—the name of the CRivVarbind.
`vl`—the value of the CRivVarbind.

3)

Constructor	<code>public CRivVarBind(String nm, CRivAtom vl)</code>
Description	Make a CRivVarBind from a name, supplied as a String, and a value.
Parameters	nm—the name of the CRivVarBind. vl—the value of the CRivVarBind.

4)

Constructor	<code>public CRivVarBind(String nm, String vl)</code>
Description	Make a CRivVarBind from a name and a value, both supplied as Strings.
Parameters	nm—the name of the CRivVarBind. vl—the value of the CRivVarBind.

5)

Constructor	<code>public CRivVarBind(String nm, int vl)</code>
Description	Make a CRivVarBind from a name, supplied as a String, and a value, supplied as an int.
Parameters	nm—the name of the CRivVarBind. vl—the value of the CRivVarBind.

6)

Constructor	<code>public CRivVarBind(String nm, long vl)</code>
Description	Make a CRivVarBind from a name, supplied as a String, and a value, supplied as a long.
Parameters	nm—the name of the CRivVarBind. vl—the value of the CRivVarBind.

Method detail

1)

Method `public boolean equals(Object obj)`

Description Compare two `CRivVarBinds` for equality. Tests if the name and value `CRivAtoms` in the `CRivVarBind` are equal using `CRivAtom.equals()`.

Parameters `obj`—the object to compare with.

Returns `true`, if this atom is the same as that specified, `false` otherwise.

Overrides `equals` in class `java.lang.Object`.

2)

Method `public CRivAtom getRVBName()`

Description Get the name of this `CRivVarBind`.

Returns A `CRivAtom` containing the name of the `CRivVarBind`.

3)

Method `public String getRVBValAsString()`

Description Get the value of this `CRivVarBind` as a `String`.

Returns A `String` containing the value of the `CRivVarBind`.

4)

Method `public CRivAtom getRVBValue()`

Description Get the value of this `CRivVarBind`.

Returns A `CRivAtom` containing the value of the `CRivVarBind`.

5)

Method `public void setRVBName(CRivAtom name)`

Description Set the name of this CRivVarBind.

Parameters `name`—a CRivAtom containing the name of the CRivVarBind.

6)

Method `public void setRVBValue(CRivAtom value)`

Description Set the value of this CRivVarBind.

Parameters `value`—a CRivAtom containing the value of the CRivVarBind.

7)

Method `public String toString()`

Description Returns a String representation of this CRivVarBind object.

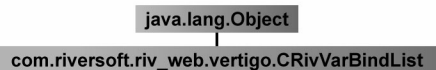
Overrides `toString` in class `java.lang.Object`

Class CRivVarBindList

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Figure 4-24 Hierarchy of class CRivVarBindList



```

public class CRivVarBindList
extends java.lang.Object
implements java.lang.Cloneable, java.io.Serializable
  
```


Description

This class provides essentially the same functionality as `java.util.Vector`, but can only contain `CRivVarBinds`. It also contains a few additional methods based on `java.util.Hashtable` which allows quick access to individual `CRivVarBinds` in the list. The list may contain more than one `CRivVarBind` with the same name.

See Also

Serialised Form

`CRivVarBind`

Constructor summary

Constructor	Reference Page
1. <code>public CRivVarBindList()</code>	158
2. <code>public CRivVarBindList(int initialCapacity)</code>	158
3. <code>public CRivVarBindList(int initialCapacity, int capacityIncrement)</code>	159

Method summary

Method	Reference Page
1. <code>public Object clone()</code>	159
2. <code>public boolean equals(Object Obj)</code>	159
3. <code>public int getRVBLCapacity()</code>	160
4. <code>public CRivVarBind getRVBLElementAt(int index)</code>	160
5. <code>public java.util.Enumeration getRVBLElements()</code>	160
6. <code>public CRivVarBind getRVBLFirstElement()</code>	161
7. <code>public int getRVBLIndexOf(CRivVarBind elem)</code>	161
8. <code>public int getRVBLIndexOf(CRivVarBind elem, int index)</code>	161
9. <code>public CRivVarBind getRVBLLastElement()</code>	162
10. <code>public int getRVBLLastIndexOf(CRivVarBind elem)</code>	162
11. <code>public int getRVBLLastIndexOf(CRivVarBind elem, int index)</code>	162
12. <code>public int getRVBLSize()</code>	163

Method	Reference Page
13. public CRivAtom <i>getRVBLValue</i> (CRivAtom nameAtom)	163
14. public CRivAtom <i>getRVBLValue</i> (String name)	163
15. public CRivVarBind <i>getRVBLVarBind</i> (CRivAtom nameAtom)	163
16. public CRivVarBind <i>getRVBLVarBind</i> (int index)	164
17. public CRivVarBind <i>getRVBLVarBind</i> (String name)	164
18. public boolean <i>isRVBLEmpty</i> ()	164
19. public boolean <i>rvblAdd</i> (CRivVarBind vb)	164
20. public void <i>rvblAdd</i> (int index, CRivVarBind element)	165
21. public boolean <i>rvblAddAll</i> (CRivVarBindList vbList)	165
22. public boolean <i>rvblAddAll</i> (int index, CRivVarBindList vbList)	165
23. public void <i>rvblAddElement</i> (CRivVarBind vb)	166
24. public void <i>rvblClear</i> ()	166
25. public boolean <i>rvblContains</i> (CRivVarBind elem)	166
26. public boolean <i>rvblContainsAll</i> (CRivVarBindList vbList)	167
27. public void <i>rvblCopyInto</i> (CRivVarBind[] anArray)	167
28. public void <i>rvblEnsureCapacity</i> (int minCapacity)	167
29. public void <i>rvblInsertElementAt</i> (CRivVarBind vb, int index)	168
30. public CRivVarBind <i>rvblPut</i> (CRivAtom name, CRivAtom value)	168
31. public boolean <i>rvblRemove</i> (CRivVarBind vb)	169
32. public CRivVarBind <i>rvblRemove</i> (int index)	169
33. public boolean <i>rvblRemoveAll</i> (CRivVarBindList vbList)	169
34. public void <i>rvblRemoveAllElements</i> ()	170
35. public boolean <i>rvblRemoveElement</i> (CRivVarBind vb)	170
36. public void <i>rvblRemoveElementAt</i> (int index)	170
37. protected void <i>rvblRemoveRange</i> (int fromIndex, int toIndex)	171

Method	Reference Page
13. public CRivAtom getRVBLValue (CRivAtom nameAtom)	163
14. public CRivAtom getRVBLValue (String name)	163
15. public CRivVarBind getRVBLVarBind (CRivAtom nameAtom)	163
16. public CRivVarBind getRVBLVarBind (int index)	164
17. public CRivVarBind getRVBLVarBind (String name)	164
18. public boolean isRVBLEmpty ()	164
19. public boolean rvblAdd (CRivVarBind vb)	164
20. public void rvblAdd (int index, CRivVarBind element)	165
21. public boolean rvblAddAll (CRivVarBindList vbList)	165
22. public boolean rvblAddAll (int index, CRivVarBindList vbList)	165
23. public void rvblAddElement (CRivVarBind vb)	166
24. public void rvblClear ()	166
25. public boolean rvblContains (CRivVarBind elem)	166
26. public boolean rvblContainsAll (CRivVarBindList vbList)	167
27. public void rvblCopyInto (CRivVarBind[] anArray)	167
28. public void rvblEnsureCapacity (int minCapacity)	167
29. public void rvblInsertElementAt (CRivVarBind vb, int index)	168
30. public CRivVarBind rvblPut (CRivAtom name, CRivAtom value)	168
31. public boolean rvblRemove (CRivVarBind vb)	169
32. public CRivVarBind rvblRemove (int index)	169
33. public boolean rvblRemoveAll (CRivVarBindList vbList)	169
34. public void rvblRemoveAllElements ()	170
35. public boolean rvblRemoveElement (CRivVarBind vb)	170
36. public void rvblRemoveElementAt (int index)	170
37. protected void rvblRemoveRange (int fromIndex, int toIndex)	171

Method	Reference Page
38. public boolean <i>rvblRetainAll</i> (CRivVarBindList vbList)	171
39. public CRivVarBind[] <i>rvblToArray</i> ()	171
40. public CRivVarBind[] <i>rvblToArray</i> (CRivVarBind[] a)	172
41. public void <i>rvblTrimToSize</i> ()	172
42. public void <i>setRVBLElementAt</i> (CRivVarBind vb, int index)	173
43. public void <i>setRVBLSize</i> (int newSize)	173
44. public CRivVarBind <i>setRVBLValueOf</i> (CRivAtom name, CRivAtom value)	174
45. public CRivVarBind <i>setRVBLVarBindAt</i> (int index, CRivVarBind element)	174
46. public String <i>toString</i> ()	174

Constructor detail

1)

Constructor	<code>public CRivVarBindList()</code>
Description	Construct an empty <code>CRivVarBindList</code> so that its internal data array has size 10 and its standard capacity increment is zero.

2)

Constructor	<code>public CRivVarBindList(int initialCapacity)</code>
Description	Construct an empty <code>CRivVarBindList</code> with the specified initial capacity and with its capacity increment equal to zero.
Parameters	<code>initialCapacity</code> —the initial capacity of the <code>CRivVarBindList</code> .
Throws	<code>java.lang.IllegalArgumentException</code> —if the specified initial capacity is negative.

3)

Constructor	<code>public CRivVarBindList(int initialCapacity, int capacityIncrement)</code>
Description	Construct an empty <code>CRivVarBindList</code> with the specified initial capacity and capacity increment. The capacity increment is the amount by which the capacity of the <code>CRivVarBind</code> is automatically incremented when its size becomes greater than its capacity. If the capacity increment is zero, the capacity is doubled each time it needs to grow.
Parameters	<code>initialCapacity</code> —the initial capacity of the <code>CRivVarBindList</code> . <code>capacityIncrement</code> —the amount by which the capacity is increased when the <code>CRivVarBindList</code> overflows.
Throws	<code>java.lang.IllegalArgumentException</code> —if the specified initial capacity is negative.

Method detail

1)

Method	<code>public Object clone()</code>
Description	Return a clone of this <code>CRivVarBindList</code> . The copy will contain a reference to a clone of the internal data array, not a reference to the original internal data array of this <code>CRivVarBindList</code> object.
Overrides	<code>clone</code> in class <code>java.lang.Object</code>
Returns	A clone of this <code>CRivVarBindList</code> .

2)

Method	<code>public boolean equals(Object obj)</code>
Description	Compare this <code>CRivVarBindList</code> to the specified object. The result is <code>true</code> if, and only if, the argument is not null and is a <code>CRivVarBindList</code> object that represents the same <code>CRivVarBindList</code> as this object, as determined by comparing the objects using the <code>toString()</code> method.
Parameters	<code>obj</code> —the object to compare with this <code>CRivVarBindList</code> for equality.

Returns true, if the objects represent the same CRivVarBindList, false, otherwise.

Overrides equals in class java.lang.Object

3)

Method `public int getRVBLCapacity()`

Description Return the current capacity of this CRivVarBindList.

Returns The current capacity (the length of its internal data array), kept in the field `elementData` of this CRivVarBindList.

4)

Method `public CRivVarBind getRVBLElementAt(int index)`

Description Return the component at the specified index.

Parameters `index`—an index into this CRivVarBindList.

Returns The component at the specified index.

Throws `ArrayIndexOutOfBoundsException`—if the index is negative or not less than the current size of this CRivVarBindList object.

See Also `getRVBLVarBind(int)`

5)

Method `public java.util.Enumeration getRVBLElements()`

Description Return an enumeration of the components of this CRivVarBindList. The returned Enumeration object will generate all items in this CRivVarBindList. The first item generated is the item at index 0, then the item at index 1, and so on.

Returns An enumeration of the components of this CRivVarBindList.

See Also `java.util.Enumeration`

6)

Method `public CRivVarBind getRVBLFirstElement()`

Description Return the first component (the item at index 0) of this CRivVarBindList.

Returns The first component of this CRivVarBindList.

Throws NoSuchElementException—if this CRivVarBindList has no components.

7)

Method `public int getRVBLIndexOf(CRivVarBind elem)`

Description Search for the first occurrence of the given argument, testing for equality using the equals method of CRivVarBind.

Parameters elem—a CRivVarBind.

Returns The index of the first occurrence of the argument in this CRivVarBindList, that is, the smallest value k such that `elem.equals(elementData[k])` is true; returns -1 if the CRivVarBind is not found.

See Also `CRivVarBind.equals(Object)`

8)

Method `public int getRVBLIndexOf(CRivVarBind elem, int index)`

Description Search for the first occurrence of the given argument, beginning the search at index, and testing for equality using the equals method of CRivVarBind.

Parameters elem—a CRivVarBind to get the index of.
index—the index to start searching from.

Returns The index of the first occurrence of the object argument in this CRivVarBindList at position index or later in the CRivVarBindList, that is, the smallest value k such that `elem.equals(elementData[k])` && `(k >= index)` is true; returns -1 if the object is not found.

See Also `CRivVarBind.equals(Object)`

9)

Method `public CRivVarBind getRVBLLastElement()`

Description Return the last component of the `CRivVarBindList`.

Returns The last component of the `CRivVarBindList`, i.e., the component at index `size() - 1`.

Throws `NoSuchElementException`—if this `CRivVarBindList` is empty.

10)

Method `public int getRVBLLastIndexOf(CRivVarBind elem)`

Description Return the index of the last occurrence of the specified `CRivVarBind` in this `CRivVarBindList`.

Parameters `elem`—the desired component.

Returns The index of the last occurrence of the specified object in this `CRivVarBindList`, that is, the largest value `k` such that `elem.equals(elementData[k])` is true; returns -1 if the `CRivVarBind` is not found.

11)

Method `public int getRVBLLastIndexOf(CRivVarBind elem, int index)`

Description Search backwards for the specified object, starting from the specified `index`, and returns an index to it.

Parameters `elem`—the `CRivVarbind` to get the index of.

`index`—the index to start searching from.

Returns The index of the last occurrence of the specified object in this `CRivVarBindList` at a position less than `index` in the `CRivVarBindList`, that is, the largest value `k`, such that `elem.equals(elementData[k]) && (k <= index)` is true; -1 if the object is not found.

12)

Method `public int getRVBLSize()`

Description Return the number of components in this CRivVarBindList.

Returns The number of components in this CRivVarBindList.

13)

Method `public CRivAtom getRVBLValue(CRivAtom nameAtom)`

Description Return the CRivAtom value of the first CRivVarBind in the list with the specified CRivAtom name, or null if there is no CRivVarBind in the list with the specified name.

Parameters nameAtom—the name of the CRivVarBind value to return.

14)

Method `public CRivAtom getRVBLValue(String name)`

Description Return the CRivAtom value of the first CRivVarBind in the list having a CRivAtom name with the specified String, or null if there is no CRivVarBind in the list with the specified name.

Parameters name—the name of the CRivVarBind value to return.

15)

Method `public CRivVarBind getRVBLVarBind(CRivAtom nameAtom)`

Description Return the first CRivVarBind in the list with the specified CRivAtom name, or null if there is no CRivVarBind in the list with the specified name.

Parameters nameAtom—the name of the CRivVarBind to return.

16)

Method `public CRivVarBind getRVBLVarBind(int index)`

Description Return the element at the specified position in this `CRivVarBindList`.

Parameters `index`—the index of the element to return.

Throws `ArrayIndexOutOfBoundsException`—if the index is out of range
`(index < 0 || index >= getRVBLSize())`.

17)

Method `public CRivVarBind getRVBLVarBind(String name)`

Description Return the first `CRivVarBind` in the list having a `CRivAtom` name with the specified `String`, or null if there is no `CRivVarBind` in the list with the specified name.

Parameters `name`—the name of the `CRivVarBind` to return.

18)

Method `public boolean isRVBLEmpty()`

Description Tests if this `CRivVarBindList` has no components.

Returns `true` if, and only if, this `CRivVarBindList` has no components, that is, its size is zero; `false` otherwise.

19)

Method `public boolean rvblAdd(CRivVarBind vb)`

Description Append the specified element to the end of this `CRivVarBindList`.

Parameters `vb`—the element to be appended to this `CRivVarBindList`.

Returns `true`, if the list changed as a result of this add.

20)

Method `public void rvblAdd(int index, CRivVarBind element)`

Description Insert the specified element at the specified position in this `CRivVarBindList`. Shifts the element currently at that position (if any) and any subsequent elements to the right (adds one to their indices).

Parameters `index`—the index at which the specified element is to be inserted.
`element`—the element to be inserted.

Throws `ArrayIndexOutOfBoundsException`—if the index is out of range
(`index < 0 || index > getRVBLSize()`).

21)

Method `public boolean rvblAddAll(CRivVarBindList vbList)`

Description Append all of the elements in the specified `CRivVarBindList` to the end of this `CRivVarBindList`, in the order that they are in the specified `CRivVarBindList`.

Parameters `vbList`—the elements to be inserted into this `CRivVarBindList`.

Returns `true`, if the list has changed as a result of this call.

22)

Method `public boolean rvblAddAll(int index, CRivVarBindList vbList)`

Description Insert all of the elements in the specified `CRivVarBindList` into this `CRivVarBindList` at the specified position. Shifts the element currently at that position (if any) and any subsequent elements to the right (increases their indices). The new elements will appear in the `CRivVarBindList` in the order that they are returned by the specified `CRivVarBindLists` iterator.

Parameters `index`—the index at which to insert the first element from the specified `CRivVarBindList`.
`vbList`—the elements to be inserted into this `CRivVarBindList`.

Returns `true`, if the list has changed as a result of this call.

Throws `ArrayIndexOutOfBoundsException`—if the index is out of range
`(index < 0 || index > getRVBLSize())`.

23)

Method `public void rvblAddElement(CRivVarBind vb)`

Description Add the specified component to the end of this `CRivVarBindList`, increasing its size by one. The capacity of this `CRivVarBindList` is increased if its size becomes greater than its capacity.

This method is identical in functionality to the `rvblAdd(CRivVarBind)` method.

Parameters `vb`—the component to be added.

See Also `rvblAdd(CRivVarBind)`

24)

Method `public void rvblClear()`

Description Remove all of the elements from this `CRivVarBindList`. The `CRivVarBindList` will be empty after this call returns (unless it throws an exception).

25)

Method `public boolean rvblContains(CRivVarBind elem)`

Description Tests if the specified `CRivVarBind` is a component in this `CRivVarBindList`.

Parameters `elem`—a `CRivVarBind` to test for inclusion in this list.

Returns `true` if, and only if, the specified `CRivVarBind` is the same as a component in this `CRivVarBindList`, as determined by the `equals` method of `CRivVarBind`. `false` otherwise.

26)

- Method** `public boolean rvblContainsAll(CRivVarBindList vbList)`
- Description** Return `true` if this `CRivVarBindList` contains all of the elements in the specified collection.
- Parameters** `vbList`—the `CRivVarBindList` to be checked for containment with this list.
- Returns** `true` if this `CRivVarBindList` contains all of the elements in the specified collection.

27)

- Method** `public void rvblCopyInto(CRivVarBind[] anArray)`
- Description** Copy the components of this `CRivVarBindList` into the specified array. The item at index `k` in this `CRivVarBindList` is copied into component `k` of `anArray`. The array must be big enough to hold all the objects in this `CRivVarBindList`, else an `IndexOutOfBoundsException` is thrown.
- Parameters** `anArray`—the array into which the components get copied.

28)

- Method** `public void rvblEnsureCapacity(int minCapacity)`
- Description** Increase the capacity of this `CRivVarBindList`, if necessary, to ensure that it can hold at least the number of components specified by the minimum capacity argument.
- If the current capacity of this `CRivVarBindList` is less than `minCapacity`, then its capacity is increased by replacing its internal data array, kept in the field `elementData`, with a larger one. The size of the new data array will be the old size plus `capacityIncrement`, unless the value of `capacityIncrement` is non-positive, in which case the new capacity will be twice the old capacity; but if this new size is still smaller than `minCapacity`, then the new capacity will be `minCapacity`.
- Parameters** `minCapacity`—the desired minimum capacity.

29)

Method `public void rvblInsertElementAt(CRivVarBind vb, int index)`

Description Insert the specified `CRivVarBind` as a component in this `CRivVarBindList` at the specified `index`. Each component in this `CRivVarBindList` with an `index` greater than, or equal, to the specified `index` is shifted upward to have an `index` one greater than the value it had previously.

The `index` must be a value greater than, or equal to, 0 and less than or equal to the current size of the `CRivVarBindList`. (If the `index` is equal to the current size of the `CRivVarBindList`, the new element is appended to the `CRivVarBindList`.)

This method is identical in functionality to the `rvblAdd(int, CRivVarBind)` method. Note that the `add` method reverses the order of the parameters, so that it more closely matches array usage.

Parameters `vb`—the component to insert.
 `index`—the position at which to insert the new component.

Throws `ArrayIndexOutOfBoundsException`—if the `index` was invalid.

See Also `getRVBLSize()`, `rvblAdd(int, CRivVarBind)`

30)

Method `public CRivVarBind rvblPut(CRivAtom name, CRivAtom value)`

Description Make a new `CRivVarBind` from the specified name and value, and add it to the list of elements. Note that unlike a `Hashtable` this can hold multiple fields with the same `CRivAtom` name.

Parameters `name`—the name of the new `CRivVarBindList` to add to the list.
 `value`—the value of the new `CRivVarBindList` to add to the list.

Returns The new `CRivVarBind` made from the given `CRivAtoms`.

31)

Method `public boolean rvblRemove(CRivVarBind vb)`

Description Remove the first occurrence of the specified element in this `CRivVarBindList`. If the `CRivVarBindList` does not contain the element, it is unchanged.

Parameters `vb`—the element to be removed from this `CRivVarBindList`, if present.

Returns `true` if the `CRivVarBindList` contained the specified element.

32)

Method `public CRivVarBind rvblRemove(int index)`

Description Remove the element at the specified position in this `CRivVarBindList`. Shifts any subsequent elements to the left (subtracts one from their indices). Returns the element that was removed from the `CRivVarBindList`.

Parameters `index`—the index of the element to be removed.

Throws `ArrayIndexOutOfBoundsException`—if the index is out of range
(`index < 0 || index > = getRVBLSize()`).

Returns The `CRivVarBind` that was removed from the list.

33)

Method `public boolean rvblRemoveAll(CRivVarBindList vbList)`

Description Remove from this `CRivVarBindList` all of its elements that are contained in the specified collection.

Parameters `vbList`—the `CRivVarBindList` that defines which elements will be removed from this list.

Returns `true` if this `CRivVarBindList` changed as a result of the call.

34)

Method `public void rvblRemoveAllElements()`

Description Remove all components from this `CRivVarBindList` and set its size to zero.

35)

Method `public boolean rvblRemoveElement(CRivVarBind vb)`

Description Remove the first (lowest-indexed) occurrence of the argument from this `CRivVarBindList`. If the `CRivVarBind` is found in this `CRivVarBindList`, each component in the `CRivVarBindList` with an index greater than, or equal to, the `CRivVarBind`'s index is shifted downward to have an index one smaller than the value it had previously.

This method is identical in functionality to the `rvblRemove(CRivVarBind)` method.

Parameters `vb`—the component to be removed.

Returns `true`, if the argument was a component of this `CRivVarBindList`. `false` otherwise.

See Also `rvblRemove(CRivVarBind)`

36)

Method `public void rvblRemoveElementAt(int index)`

Description Delete the component at the specified `index`. Each component in this `CRivVarBindList` with an index greater than, or equal to, the specified `index` is shifted downward to have an index one smaller than the value it had previously. The size of this `CRivVarBindList` is decreased by 1.

The index must be a value greater than, or equal, to 0 and less than the current size of the `CRivVarBindList`.

This method is identical in functionality to the `rvblRemove(int)` method.

The `rvblRemove(int)` method returns the old value that was stored at the specified position.

Parameters `index`—the index of the object to remove.

Throws `ArrayIndexOutOfBoundsException`—if the index was invalid.

See Also `getRVBLSize()`, `rvblRemove(int)`

37)

Method `protected void rvblRemoveRange(int fromIndex, int toIndex)`

Description Remove from this list all of the elements whose index is between `fromIndex`, inclusive and `toIndex`, exclusive. Shifts any succeeding elements to the left (reduces their index). This call shortens the `CRivVarBind` by `(toIndex - fromIndex)` elements. If `toIndex==fromIndex`, this operation has no effect.

Parameters `fromIndex`—the index of the first element to be removed.
`toIndex`—the index after the last element to be removed.

38)

Method `public boolean rvblRetainAll(CRivVarBindList vbList)`

Description Retain only the elements in this `CRivVarBindList` that are contained in the specified list. In other words, this method removes from this `CRivVarBindList` all of its elements that are not contained in the specified `CRivVarBindList`.

Parameters `vbList`—the `CRivVarBindList` that defines which elements this list will retain.

Returns `true`, if this `CRivVarBindList` changed as a result of the call.

39)

Method `public CRivVarBind[] rvblToArray()`

Description Return an array containing all of the elements in this `CRivVarBindList` in the correct order. This is an array of `CRivVarBinds`.

Returns An array containing the `CRivVarBinds` in this list.

40)

Method `public CRivVarBind[] rvblToArray(CRivVarBind[] a)`

Description Return an array containing all of the elements in this `CRivVarBindList` in the correct order. If the `CRivVarBindList` fits in the specified array, it is returned therein. Otherwise, a new array is allocated with the size of this `CRivVarBindList`.

If the `CRivVarBindList` fits in the specified array with room to spare (i.e., the array has more elements than the `CRivVarBindList`), the element in the array immediately following the end of the `CRivVarBindList` is set to null. This is useful in determining the length of the `CRivVarBindList` only if the caller knows that the `CRivVarBindList` does not contain any null elements.

Parameters `a`—the array into which the elements of the `CRivVarBindList` are to be stored, if it is big enough; otherwise, a new array of the same runtime type is allocated for this purpose.

Returns An array containing the elements of the `CRivVarBindList`.

41)

Method `public void rvblTrimToSize()`

Description Trim the capacity of this `CRivVarBindList` to be the `CRivVarBindList`'s current size. If the capacity of this `CRivVarBindList` is larger than its current size, then the capacity is changed to equal the size by replacing its internal data array, kept in the field `elementData`, with a smaller one. An application can use this operation to minimise the storage of a `CRivVarBindList`.

42)

Method `public void setRVBLElementAt(CRivVarBind vb, int index)`

Description Set the component at the specified `index` of this `CRivVarBindList` to be the specified `CRivVarBind`. The previous component at that position is discarded.

The `index` must be a value greater than, or equal to, 0 and less than the current size of the `CRivVarBindList`.

This method is identical in functionality to the `setRVBLVarBindAt` method. Note that the `setRVBLVarBindAt` method reverses the order of the parameters, to more closely match array usage. Note, also, that the `setRVBLVarBindAt` method returns the old value that was stored at the specified position.

Parameters `vb`—the `CRivVarBind` the component is to be set to.
`index`—the specified `index`.

Throws `ArrayIndexOutOfBoundsException`—if the `index` was invalid.

See Also `getRVBLSize()`, `setRVBLVarBindAt(int index, CRivVarBind element)`

43)

Method `public void setRVBLSize(int newSize)`

Description Set the size of this `CRivVarBindList`. If the new size is greater than the current size, new null items are added to the end of the `CRivVarBindList`. If the new size is less than the current size, all components at `index newSize` and greater are discarded.

Parameters `newSize`—the new size of this `CRivVarBindList`.

Throws `ArrayIndexOutOfBoundsException`—if `newSize` is negative.

44)

Method	<code>public CRivVarBind setRVBLValueOf(CRivAtom name, CRivAtom value)</code>
Description	Set the value of the <code>CRivVarBind</code> having the given name, to have the supplied value. For lists which contain multiple keys, only the first entry which has the given name will have its value set.
Parameters	<code>name</code> —the name of the <code>CRivVarBind</code> to set the value of. <code>value</code> —the value the <code>CRivVarBind</code> will take.
Returns	The new <code>CRivVarBind</code> made from the given <code>CRivAtoms</code> , or <code>null</code> if the table did not contain a <code>CRivVarBind</code> with the specified name key.

45)

Method	<code>public CRivVarBind setRVBLVarBindAt(int index, CRivVarBind element)</code>
Description	Replace the element at the specified position in this <code>CRivVarBindList</code> with the specified element.
Parameters	<code>index</code> —the index of the element to replace. <code>element</code> —the element to be stored at the specified position.
Returns	The element previously at the specified position.
Throws	<code>ArrayIndexOutOfBoundsException</code> —index out of range (<code>index < 0 index >= getRVBLSize()</code>).

46)

Method	<code>public synchronized String toString()</code>
Description	Return a <code>String</code> representation of this <code>CRivVarBindList</code> , containing the <code>String</code> representation of each element.
Overrides	<code>toString</code> in class <code>java.lang.Object</code>

Interface IRivAlgebraic

(com.riversoft.riv_web.vertigo)

Description

This interface defines the constants representing the different Boolean algebraic operators.

Hierarchy

public interface *IRivAlgebraic*

Field summary

Field	Reference Page
1. public static final int <i>RA_AND</i>	175
2. public static final int <i>RA_NO_OP</i>	175
3. public static final int <i>RA_NOT</i>	176
4. public static final int <i>RA_OR</i>	176

Field detail

1)

Field `public static final int RA_AND`

Description Constant representing a Boolean AND.

Value 0

2)

Field `public static final int RA_NO_OP`

Description Constant indicating that no operator has been defined.

Value 3

3)

Field	<code>public static final int RA_NOT</code>
Description	Constant representing a Boolean NOT.
Value	2

4)

Field	<code>public static final int RA_OR</code>
Description	Constant representing a Boolean OR.
Value	1

Interface IRivConstants

(`com.riversoft.riv_web.vertigo`)

All known implementing classes

`CRivClientHelper`

Description

This interface contains a set of constants for use with MWFM applications. It contains constants for attributes such as the standard field names for event records in `riv_f_amos`, model records in `riv_model`, or user and profile records in `riv_auth`. It also holds values for some of these fields, such as the possible values for the `Severity`, `InternalAction` and `EventType` fields in event records.

Hierarchy

```
public interface IRivConstants
```

See also

`CRivRecord`

Field summary

Field	Reference Page
1. <code>public static final int REC_ACK</code>	180
2. <code>public static final int REC_ALERT</code>	180
3. <code>public static final int REC_ASSIGN</code>	180
4. <code>public static final int REC_CLEAR</code>	181
5. <code>public static final int REC_CLEAR_ALL</code>	181
6. <code>public static final int REC_CLEAR_CHAIN</code>	181
7. <code>public static final int REC_DATA</code>	181
8. <code>public static final int REC_DEASSIGN</code>	181
9. <code>public static final int REC_DELETE</code>	182
10. <code>public static final int REC_EVENT</code>	182
11. <code>public static final int REC_NEW</code>	182
12. <code>public static final int REC_NONE</code>	182
13. <code>public static final int REC_TOOL</code>	183
14. <code>public static final int REC_UNACK</code>	183
15. <code>public static final int REC_UNDEFINED_ACTION</code>	183
16. <code>public static final int REC_UPDATE</code>	183
17. <code>public static final String RIV_AMOS_DN_CLAS</code>	183
18. <code>public static final String RIV_AMOS_DN_MOJO</code>	184
19. <code>public static final String RIV_AMOS_DN_TOPO</code>	184
20. <code>public static final String RIV_AMOS_DN_WHIP</code>	184
21. <code>public static final String RIV_AMOS_FN_ACKED</code>	184
22. <code>public static final String RIV_AMOS_FN_ACTN</code>	184
23. <code>public static final String RIV_AMOS_FN_ACTVE</code>	185
24. <code>public static final String RIV_AMOS_FN_ASSGN</code>	185
25. <code>public static final String RIV_AMOS_FN_CHTME</code>	185
26. <code>public static final String RIV_AMOS_FN_CONTACT</code>	185
27. <code>public static final String RIV_AMOS_FN_CONTO</code>	185
28. <code>public static final String RIV_AMOS_FN_CORRID</code>	186
29. <code>public static final String RIV_AMOS_FN_CRTME</code>	186
30. <code>public static final String RIV_AMOS_FN_EGRPID</code>	186
31. <code>public static final String RIV_AMOS_FN_EVID</code>	186

Field	Reference Page
32. public static final String <i>RIV_AMOS_FN_EVTYPE</i>	186
33. public static final String <i>RIV_AMOS_FN_GLYPH</i>	187
34. public static final String <i>RIV_AMOS_FN_INTAC</i>	187
35. public static final String <i>RIV_AMOS_FN_NAME</i>	187
36. public static final String <i>RIV_AMOS_FN_OBJID</i>	187
37. public static final String <i>RIV_AMOS_FN_OCCRD</i>	187
38. public static final String <i>RIV_AMOS_FN_PART</i>	188
39. public static final String <i>RIV_AMOS_FN_SEVTY</i>	188
40. public static final String <i>RIV_AMOS_TN_ACLS</i>	188
41. public static final String <i>RIV_AMOS_TN_CONTS</i>	188
42. public static final String <i>RIV_AMOS_TN_ENTITY</i>	189
43. public static final String <i>RIV_AMOS_TN_EVENTS</i>	189
44. public static final String <i>RIV_AUTH_DB_EVENTS_OBJECT</i>	189
45. public static final String <i>RIV_AUTH_DB_PASSWORD</i>	189
46. public static final String <i>RIV_AUTH_DB_PERMISSIONS</i>	190
47. public static final String <i>RIV_AUTH_DB_PROFILE</i>	190
48. public static final String <i>RIV_AUTH_DB_PROFILES_TBL</i>	190
49. public static final String <i>RIV_AUTH_DB_RANK</i>	190
50. public static final String <i>RIV_AUTH_DB_USERNAME</i>	190
51. public static final String <i>RIV_AUTH_DB_USERS_OBJECT</i>	191
52. public static final String <i>RIV_AUTH_DB_USERS_TBL</i>	191
53. public static final String <i>RIV_AUTH_FIELD_CHANGE</i>	191
54. public static final String <i>RIV_AUTH_FIELD_CREATE</i>	191
55. public static final String <i>RIV_AUTH_FIELD_DELETE</i>	191
56. public static final String <i>RIV_AUTH_FIELD_EV_ACK</i>	192
57. public static final String <i>RIV_AUTH_FIELD_EV_ASSIGN</i>	192

Field	Reference Page
58. public static final String <i>RIV_AUTH_FIELD_EV_CLEAR</i>	192
59. public static final String <i>RIV_AUTH_FIELD_EV_DEASSIGN</i>	192
60. public static final String <i>RIV_AUTH_FIELD_EV_UNACK</i>	193
61. public static final String <i>RIV_AUTH_FIELD_VIEW</i>	193
62. public static final String <i>RIV_AUTH_PASSWORD</i>	193
63. public static final String <i>RIV_AUTH_QUERY</i>	193
64. public static final int <i>RIV_AUTH_RANK_HIGHEST</i>	193
65. public static final int <i>RIV_AUTH_RANK_LOWEST</i>	194
66. public static final String <i>RIV_AUTH_USERNAME</i>	194
67. public static final String <i>RIV_MDL_DN_MSTR</i>	194
68. public static final String <i>RIV_MDL_DN_TRANS</i>	194
69. public static final String <i>RIV_MDL_FN_ACTN</i>	195
70. public static final String <i>RIV_MDL_FN_ACTVE</i>	195
71. public static final String <i>RIV_MDL_FN_ADDR</i>	195
72. public static final String <i>RIV_MDL_FN_CHTME</i>	195
73. public static final String <i>RIV_MDL_FN_CLASS</i>	195
74. public static final String <i>RIV_MDL_FN_CONTO</i>	196
75. public static final String <i>RIV_MDL_FN_CRTME</i>	196
76. public static final String <i>RIV_MDL_FN_DESCR</i>	196
77. public static final String <i>RIV_MDL_FN_EOID</i>	196
78. public static final String <i>RIV_MDL_FN_ETYPE</i>	196
79. public static final String <i>RIV_MDL_FN_NAME</i>	197
80. public static final String <i>RIV_MDL_FN_OBJID</i>	197
81. public static final String <i>RIV_MDL_FN_PART</i>	197
82. public static final String <i>RIV_MDL_FN_RELTO</i>	197
83. public static final String <i>RIV_MDL_FN_SECTY</i>	197
84. public static final String <i>RIV_MDL_FN_STATS</i>	198
85. public static final String <i>RIV_MDL_TN_CONTS</i>	198
86. public static final String <i>RIV_MDL_TN_EBNR</i>	198
87. public static final String <i>RIV_MDL_TN_ENTITY</i>	198
88. public static final int <i>SEV_CLEAR</i>	199
89. public static final int <i>SEV_CRITICAL</i>	199
90. public static final int <i>SEV_MAJOR</i>	199

Field	Reference Page
91. <code>public static final int SEV_MINOR</code>	199
92. <code>public static final int SEV_NO_SEV</code>	199
93. <code>public static final int SEV_UNKNOWN</code>	200
94. <code>public static final int SEV_WARNING</code>	200

Field detail

1)

Field	<code>public static final int REC_ACK</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that the event has been acknowledged.
Value	1

2)

Field	<code>public static final int REC_ALERT</code>
Description	The value of the <code>EventType</code> field in a <code>CRivRecord</code> for an event which indicates that this is an "Alert", a network occurrence that requires the attention of a network administrator.
Value	2

3)

Field	<code>public static final int REC_ASSIGN</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that the event has been assigned.
Value	3

4)

Field	<code>public static final int REC_CLEAR</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that the event has been cleared.
Value	6

5)

Field	<code>public static final int REC_CLEAR_ALL</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that this event has been cleared with all related events.
Value	7

6)

Field	<code>public static final int REC_CLEAR_CHAIN</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that this event has been cleared with all correlated events.
Value	8

7)

Field	<code>public static final int REC_DATA</code>
Description	The value of the <code>EventType</code> field in a <code>CRivRecord</code> for an event which indicates that this is an item of information.
Value	1

8)

Field	<code>public static final int REC_DEASSIGN</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that the event has been deassigned.
Value	4

9)

Field	<code>public static final int REC_DELETE</code>
Description	The value of the <code>ActionType</code> field in a <code>CRivRecord</code> which indicates that this is a deletion of a record.
Value	2

10)

Field	<code>public static final int REC_EVENT</code>
Description	The value of the <code>EventType</code> field in a <code>CRivRecord</code> for an event which indicates that this is an "Event", an occurrence on the network.
Value	0

11)

Field	<code>public static final int REC_NEW</code>
Description	The value of the <code>ActionType</code> field in a <code>CRivRecord</code> which indicates that this is a new record.
Value	0

12)

Field	<code>public static final int REC_NONE</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that no action has been done on the event.
Value	0

13)

Field	<code>public static final int REC_TOOL</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that a tool has been launched for the event.
Value	5

14)

Field	<code>public static final int REC_UNACK</code>
Description	The value of the <code>InternalAction</code> field in a <code>CRivRecord</code> for an event which indicates that the event has been unacknowledged.
Value	2

15)

Field	<code>public static final int REC_UNDEFINED_ACTION</code>
Description	The value of the <code>ActionType</code> field in a <code>CRivRecord</code> which indicates that the action type is undefined.
Value	3

16)

Field	<code>public static final int REC_UPDATE</code>
Description	The value of the <code>ActionType</code> field in a <code>CRivRecord</code> which indicates that this is an update to a record.
Value	1

17)

Field	<code>public static final String RIV_AMOS_DN_CLAS</code>
Description	The name of the class database held in <code>riv_f_amos</code> . This holds instantiation information, super class information, data dictionary details and details of the extensions.
Value	"class"

18)

Field	<code>public static final String RIV_AMOS_DN_MOJO</code>
Description	The name of the master object journal database of <code>riv_f_amos</code> .
Value	“mojo”

19)

Field	<code>public static final String RIV_AMOS_DN_TOPO</code>
Description	The name of the <code>topoCache</code> database of <code>riv_f_amos</code> which holds a cache of the master topology.
Value	“topoCache”

20)

Field	<code>public static final String RIV_AMOS_DN_WHIP</code>
Description	The name of the template whippet database in <code>riv_f_amos</code> .
Value	“whippet”

21)

Field	<code>public static final String RIV_AMOS_FN_ACKED</code>
Description	The <code>riv_f_amos</code> column name to denote whether the event has been acknowledged or unacknowledged.
Value	“Acknowledged”

22)

Field	<code>public static final String RIV_AMOS_FN_ACTN</code>
Description	The <code>riv_f_amos</code> column name for the type of the event, which is used to indicate whether it is a new event, an update, or a delete of an existing event.
Value	“ActionType”

23)

Field	<code>public static final String RIV_AMOS_FN_ACTVE</code>
Description	The <code>riv_f_amos</code> column name in the <code>entityByName</code> table for the flag which indicates whether an Active Object Class is needed.
Value	“IsActive”

24)

Field	<code>public static final String RIV_AMOS_FN_ASSGN</code>
Description	The <code>riv_f_amos</code> column name which names the person that the event has been assigned to.
Value	“AssignedTo”

25)

Field	<code>public static final String RIV_AMOS_FN_CHTME</code>
Description	The <code>riv_f_amos</code> column name for the time of the last occurrence of the event.
Value	“ChangeTime”

26)

Field	<code>public static final String RIV_AMOS_FN_CONTACT</code>
Description	The <code>riv_f_amos</code> column name which names the contact group responsible for the device that generated the event.
Value	“Contact”

27)

Field	<code>public static final String RIV_AMOS_FN_CONTO</code>
Description	The <code>riv_f_amos</code> column name in the <code>entityByName</code> table which holds the list of elements or other containers contained by an object.
Value	“Contains”

28)

Field	<code>public static final String RIV_AMOS_FN_CORRID</code>
Description	The <code>riv_f_amos</code> column name for the list of IDs of associated events.
Value	“CorrelatedId”

29)

Field	<code>public static final String RIV_AMOS_FN_CRTME</code>
Description	The <code>riv_f_amos</code> column name for the time of the first occurrence of the event.
Value	“CreateTime”

30)

Field	<code>public static final String RIV_AMOS_FN_EGRPID</code>
Description	The <code>riv_f_amos</code> column name for the group identifier of the event, set when an event is correlated into an alert.
Value	“EventGroupId”

31)

Field	<code>public static final String RIV_AMOS_FN_EVID</code>
Description	The <code>riv_f_amos</code> column name for the unique ID of an event.
Value	“EventId”

32)

Field	<code>public static final String RIV_AMOS_FN_EVTTYPE</code>
Description	The <code>riv_f_amos</code> column name which lists the type of the event : event, data or alert.
Value	“EventType”
See Also	<code>REC_EVENT</code> , <code>REC_DATA</code> , <code>REC_ALERT</code>

33)

Field	<code>public static final String RIV_AMOS_FN_GLYPH</code>
Description	The <code>riv_f_amos</code> column name for an alert display glyph.
Value	“ActionGlyph”

34)

Field	<code>public static final String RIV_AMOS_FN_INTAC</code>
Description	The <code>riv_f_amos</code> column name for the internal action associated with the event. This is used to indicate whether the event has been acknowledged, unacknowledged, assigned, cleared or a tool has been launched.
Value	“InternalAction”

35)

Field	<code>public static final String RIV_AMOS_FN_NAME</code>
Description	The <code>riv_f_amos</code> column name for the name of the Active Object associated with the event.
Value	“EntityName”

36)

Field	<code>public static final String RIV_AMOS_FN_OBJID</code>
Description	The <code>riv_f_amos</code> column name in the <code>entityByName</code> table for the unique Object ID of a network entity.
Value	“ObjectId”

37)

Field	<code>public static final String RIV_AMOS_FN_OCCRD</code>
Description	The <code>riv_f_amos</code> column name for the number of identical events that have occurred.
Value	“Occurred”

38)

Field	<code>public static final String RIV_AMOS_FN_PART</code>
Description	The <code>riv_f_amos</code> PartOfName column name for an event.
Value	“PartOfName”

39)

Field	<code>public static final String RIV_AMOS_FN_SEVTY</code>
Description	The <code>riv_f_amos</code> column name for the OSI severity code : clear, unknown, warning, minor, major, critical or none.
Value	“Severity”

40)

Field	<code>public static final String RIV_AMOS_TN_ACLS</code>
Description	The name of the active classes table in the class database held by <code>riv_f_amos</code> .
Value	“activeClasses”

41)

Field	<code>public static final String RIV_AMOS_TN_CONTS</code>
Description	The name of the containers table in the topology cache database in <code>riv_f_amos</code> . This uses the containment model principle to consider each network entity as being contained by other network entities. It is automatically populated as a result of entries made into the <code>entityByName</code> table.
Value	“containers”
See Also	<code>RIV_AMOS_TN_ENTITY</code>

42)

Field	<code>public static final String RIV_AMOS_TN_ENTITY</code>
Description	The name of the <code>entityByName</code> table in the topology cache database in <code>riv_f_amos</code> , which is used to hold information about all the discovered entities on the network.
Value	“entityByName”

43)

Field	<code>public static final String RIV_AMOS_TN_EVENTS</code>
Description	The name of the events table in the master object journal of <code>riv_f_amos</code> which holds the master record of the events.
Value	“events”

44)

Field	<code>public static final String RIV_AUTH_DB_EVENTS_OBJECT</code>
Description	The label for the events category in the permission part of a profile in <code>riv_auth</code> . This lists whether the profile allows events to be assigned, deassigned, acknowledged, unacknowledged, or cleared.
Value	“Events”

45)

Field	<code>public static final String RIV_AUTH_DB_PASSWORD</code>
Description	The column name for the user’s password in the <code>auth.users</code> table of <code>riv_auth</code> .
Value	“Password”

46)

Field	<code>public static final String RIV_AUTH_DB_PERMISSIONS</code>
Description	The column name for the permissions part of a profile in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	“Permissions”

47)

Field	<code>public static final String RIV_AUTH_DB_PROFILE</code>
Description	The column name for the profile's name in the <code>auth.users</code> or <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	“Profile”

48)

Field	<code>public static final String RIV_AUTH_DB_PROFILES_TBL</code>
Description	The name of the profiles table of <code>riv_auth</code> , which contains information about a user's rank relative to other users and a list of permissions available for that user.
Value	“auth.profiles”

49)

Field	<code>public static final String RIV_AUTH_DB_RANK</code>
Description	The column name for the rank of a profile in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	“Rank”

50)

Field	<code>public static final String RIV_AUTH_DB_USERNAME</code>
Description	The column name for the user's name in the <code>auth.users</code> table of <code>riv_auth</code> .
Value	“Name”

51)

Field	<code>public static final String RIV_AUTH_DB_USERS_OBJECT</code>
Description	The label for the user's category in the permission part of a profile in <code>riv_auth</code> . This gives lists of whether users and profiles in <code>riv_auth</code> may be viewed, changed, created or deleted.
Value	"Users"

52)

Field	<code>public static final String RIV_AUTH_DB_USERS_TBL</code>
Description	The name of the user's table in <code>riv_auth</code> , which stores the name, password and profile of a user.
Value	"auth.users"

53)

Field	<code>public static final String RIV_AUTH_FIELD_CHANGE</code>
Description	The field name of the change action for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pChange"

54)

Field	<code>public static final String RIV_AUTH_FIELD_CREATE</code>
Description	The field name of the create action for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pCreate"

55)

Field	<code>public static final String RIV_AUTH_FIELD_DELETE</code>
Description	The field name of the delete action for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pDelete"

56)

Field	<code>public static final String RIV_AUTH_FIELD_EV_ACK</code>
Description	The field name of the acknowledge action in the Events category for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pAck"

57)

Field	<code>public static final String RIV_AUTH_FIELD_EV_ASSIGN</code>
Description	The field name of the assign action in the Events category for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pAssign"

58)

Field	<code>public static final String RIV_AUTH_FIELD_EV_CLEAR</code>
Description	The field name of the clear action in the Events category for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pClear"

59)

Field	<code>public static final String RIV_AUTH_FIELD_EV_DEASSIGN</code>
Description	The field name of the de-assign action in the Events category for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pDeAssign"

60)

Field	<code>public static final String RIV_AUTH_FIELD_EV_UNACK</code>
Description	The field name of the unacknowledge action in the Events category for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pUnAck"

61)

Field	<code>public static final String RIV_AUTH_FIELD_VIEW</code>
Description	The field name of the view action for a user's permissions in the <code>auth.profiles</code> table of <code>riv_auth</code> .
Value	"pView"

62)

Field	<code>public static final String RIV_AUTH_PASSWORD</code>
Description	The field name for a user's password used in encoded queries to <code>riv_auth</code> .
Value	"RivAuthPassword"

63)

Field	<code>public static final String RIV_AUTH_QUERY</code>
Description	The field name for a user's query used in encoded queries to <code>riv_auth</code> .
Value	"RivAuthQuery"

64)

Field	<code>public static final int RIV_AUTH_RANK_HIGHEST</code>
Description	The value of the rank field in the <code>auth.profiles</code> table of <code>riv_auth</code> which indicates the highest possible ranking.
Value	0

65)

Field	<code>public static final int RIV_AUTH_RANK_LOWEST</code>
Description	The value of the rank field in the <code>auth.profiles</code> table of <code>riv_auth</code> which indicates the lowest possible ranking.
Value	50

66)

Field	<code>public static final String RIV_AUTH_USERNAME</code>
Description	The field name for a user's name used in encoded queries to <code>riv_auth</code> .
Value	"RivAuthUserName"

67)

Field	<code>public static final String RIV_MDL_DN_MSTR</code>
Description	The name of the master database in <code>riv_model</code> . The master database is responsible for holding all the network entities, their containment and their connections. It consists of three tables, <code>entityByName</code> , <code>entityByNeighbor</code> and <code>containers</code> .
Value	"master"
See Also	<code>RIV_MDL_TN_ENTITY</code> , <code>RIV_MDL_TN_EBNR</code> , <code>RIV_MDL_TN_CONTS</code>

68)

Field	<code>public static final String RIV_MDL_DN_TRANS</code>
Description	The name of the translations database in <code>riv_model</code> . This is responsible for holding <code>integer-to-String</code> mappings for externally-defined data types used by the master database. It consists of as many tables as there are external data types. By default there are three tables, <code>entityTypes</code> , <code>boolean</code> and <code>actions</code> .
Value	"translations"
See Also	<code>RIV_MDL_DN_MSTR</code>

69)

Field	<code>public static final String RIV_MDL_FN_ACTN</code>
Description	The <code>riv_model</code> column name for the type of the record that is used to indicate whether it is a new record, an update, or a delete of an existing record.
Value	“ActionType”

70)

Field	<code>public static final String RIV_MDL_FN_ACTVE</code>
Description	The <code>riv_model</code> column name for the flag indicating whether an Active Object Class is needed.
Value	“IsActive”

71)

Field	<code>public static final String RIV_MDL_FN_ADDR</code>
Description	The <code>riv_model</code> column name for the list of OSI model Layer 1-7 addresses for the entity.
Value	“Address”

72)

Field	<code>public static final String RIV_MDL_FN_CHTME</code>
Description	The <code>riv_model</code> column name for the time of the last modification to the network entity record.
Value	“ChangeTime”

73)

Field	<code>public static final String RIV_MDL_FN_CLASS</code>
Description	The <code>riv_model</code> column name for the class name of the entity.
Value	“ClassName”

74)

Field	<code>public static final String RIV_MDL_FN_CONTO</code>
Description	The riv_model column name for the list of elements or other containers contained by this entity.
Value	“Contains”

75)

Field	<code>public static final String RIV_MDL_FN_CRTME</code>
Description	The riv_model column name for the creation time of the network entity record in the table.
Value	“CreateTime”

76)

Field	<code>public static final String RIV_MDL_FN_DESCR</code>
Description	The riv_model column name for the value of the <code>sysDescr</code> MIB variable, or other suitable description, of the entity.
Value	“Description”

77)

Field	<code>public static final String RIV_MDL_FN_EOID</code>
Description	The riv_model column name for the value of the <code>sysOID</code> MIB variable of the entity.
Value	“EntityOID”

78)

Field	<code>public static final String RIV_MDL_FN_ETYPE</code>
Description	The riv_model column name for the element type of the entity.
Value	“EntityType”

79)

Field	<code>public static final String RIV_MDL_FN_NAME</code>
Description	The <code>riv_model</code> column name for the unique descriptive name of a network entity.
Value	“EntityName”

80)

Field	<code>public static final String RIV_MDL_FN_OBJID</code>
Description	The <code>riv_model</code> column name for the unique Object ID of a network entity.
Value	“ObjectId”

81)

Field	<code>public static final String RIV_MDL_FN_PART</code>
Description	The <code>riv_model</code> PartOfName column name for a network entity.
Value	“PartOfName”

82)

Field	<code>public static final String RIV_MDL_FN_RELTO</code>
Description	The <code>riv_model</code> column name for the list of connections to the network entity.
Value	“RelatedTo”

83)

Field	<code>public static final String RIV_MDL_FN_SECTY</code>
Description	The <code>riv_model</code> column name for the password needed to access the network entity, if applicable.
Value	“Security”

84)

Field	<code>public static final String RIV_MDL_FN_STATS</code>
Description	The <code>riv_model</code> column name for the flag showing the status of the network entity.
Value	“Status”

85)

Field	<code>public static final String RIV_MDL_TN_CONTS</code>
Description	The name of the containers table in the master database of <code>riv_model</code> . This uses the Containment Model principle to consider each network entity as being contained by other network entities. It is automatically populated as a result of entries made into the <code>entityByName</code> table.
Value	“containers”
See Also	<code>RIV_MDL_TN_ENTITY</code>

86)

Field	<code>public static final String RIV_MDL_TN_EBNR</code>
Description	The name of the <code>entityByNeighbor</code> table in the master database of <code>riv_model</code> , which holds all the connectivity information pertaining to each network entity. An entry is made in the table for every link in the topology, though the level of detailed information about the connections is optional.
Value	“entityByNeighbour”

87)

Field	<code>public static final String RIV_MDL_TN_ENTITY</code>
Description	The name of the <code>entityByName</code> table in the master database of <code>riv_model</code> , which is used to hold information about all the discovered entities on the network.
Value	“entityByName”

88)

Field	<code>public static final int SEV_CLEAR</code>
Description	The value of the <code>Severity</code> field in a <code>CRivRecord</code> for an event that indicates that the severity of the event is clear.
Value	0

89)

Field	<code>public static final int SEV_CRITICAL</code>
Description	The value of the <code>Severity</code> field in a <code>CRivRecord</code> for an event that indicates that the event is of critical severity.
Value	5

90)

Field	<code>public static final int SEV_MAJOR</code>
Description	The value of the <code>Severity</code> field in a <code>CRivRecord</code> for an event that indicates that the event is of major severity.
Value	4

91)

Field	<code>public static final int SEV_MINOR</code>
Description	The value of the <code>Severity</code> field in a <code>CRivRecord</code> for an event that indicates that the event is of minor severity.
Value	3

92)

Field	<code>public static final int SEV_NO_SEV</code>
Description	The value of the <code>Severity</code> field in a <code>CRivRecord</code> for an event that indicates that the event has no severity.
Value	6

93)

Field `public static final int SEV_UNKNOWN`Description The value of the `Severity` field in a `CRivRecord` for an event that indicates that the severity of the event is unknown.

Value 1

94)

Field `public static final int SEV_WARNING`Description The value of the `Severity` field in a `CRivRecord` for an event that indicates that the severity of the event is a 'warning'.

Value 2

Interface IRivDataType

`(com.riversoft.riv_web.vertigo)`

All known implementing classes

`CRivAtom`

Description

This interface defines the constants representing the different data types supported by `CRivAtom`.

Hierarchy

`public interface IRivDataType`

See Also

`CRivAtom`

Field summary

Field	Reference Page
1. <code>public static final int RDT_ASN1</code>	201
2. <code>public static final int RDT_BYTE_ARRAY</code>	201
3. <code>public static final int RDT_FLOAT</code>	201
4. <code>public static final int RDT_INTEGER</code>	202
5. <code>public static final int RDT_LIST</code>	202
6. <code>public static final int RDT_LONG</code>	202
7. <code>public static final int RDT_OBJECT</code>	202
8. <code>public static final int RDT_STRING</code>	202
9. <code>public static final int RDT_UNDEFINED</code>	203

Field detail

1)

Field	<code>public static final int RDT_ASN1</code>
Description	A <code>CRivAtom</code> constructed from a <code>CRivASN1Address</code> , that has type <code>RDT_ASN1</code> .
Value	5

2)

Field	<code>public static final int RDT_BYTE_ARRAY</code>
Description	A <code>CRivAtom</code> constructed from a <code>byte []</code> , that has type <code>RDT_BYTE_ARRAY</code> .
Value	4

3)

Field	<code>public static final int RDT_FLOAT</code>
Description	A <code>CRivAtom</code> constructed from a <code>float</code> , that has type <code>RDT_FLOAT</code> .
Value	3

4)

Field	<code>public static final int RDT_INTEGER</code>
Description	A <code>CRivAtom</code> constructed from an <code>int</code> , that has type <code>RDT_INTEGER</code> .
Value	0

5)

Field	<code>public static final int RDT_LIST</code>
Description	A <code>CRivAtom</code> constructed from a <code>java.util.Vector</code> of <code>CRivAtom</code> objects that has type <code>RDT_LIST</code> .
Value	6

6)

Field	<code>public static final int RDT_LONG</code>
Description	A <code>CRivAtom</code> constructed from a <code>long</code> , that has type <code>RDT_LONG</code> .
Value	1

7)

Field	<code>public static final int RDT_OBJECT</code>
Description	A <code>CRivAtom</code> constructed from a <code>CRivVarBindList</code> that has type <code>RDT_OBJECT</code> .
Value	7

8)

Field	<code>public static final int RDT_STRING</code>
Description	A <code>CRivAtom</code> constructed from a <code>String</code> , that has type <code>RDT_STRING</code> .
Value	2

9)

Field	<code>public static final int RDT_UNDEFINED</code>
Description	A <code>CRivAtom</code> of undefined type, or constructed from a null value.
Value	8

Interface IRivNodeType

(`com.riversoft.riv_web.vertigo`)

Description

This interface defines the constants representing the different types of nodes in a tree such as a branch, leaf, or root.

Hierarchy

public interface *IRivNodeType*

Field summary

Field	Reference Page
1. <code>public static final int RNT_BRANCH</code>	203
2. <code>public static final int RNT_LEAF</code>	204
3. <code>public static final int RNT_ROOT</code>	204

Field detail

1)

Field	<code>public static final int RNT_BRANCH</code>
Description	Constant indicating that the node represents a branch.
Value	1

2)

Field `public static final int RNT_LEAF`

Description Constant indicating that the node represents a leaf.

Value 2

3)

Field `public static final int RNT_ROOT`

Description Constant indicating that the node is the root node of the tree.

Value 0

Interface IRivOper

`(com.riversoft.riv_web.vertigo)`

All Known Implementing Classes

`CRivAtom`

Description

This interface defines the constants representing the different operations that can be performed on a tree.

Hierarchy

`public interface IRivOper`

Field summary

Field	Reference Page
1. <code>public static final int RO_BETWEEN</code>	205
2. <code>public static final int RO_EQUALS</code>	205
3. <code>public static final int RO_EXISTS</code>	206
4. <code>public static final int RO_GREATER_THAN</code>	206
5. <code>public static final int RO_GREATER_THAN_OR_EQUAL</code>	206
6. <code>public static final int RO_IN</code>	206
7. <code>public static final int RO_IS_NOT_NULL</code>	206
8. <code>public static final int RO_IS_NULL</code>	207
9. <code>public static final int RO_LESS_THAN</code>	207
10. <code>public static final int RO_LESS_THAN_OR_EQUAL</code>	207
11. <code>public static final int RO_MATCHES</code>	207
12. <code>public static final int RO_NONE</code>	207
13. <code>public static final int RO_NOT_BETWEEN</code>	208
14. <code>public static final int RO_NOT_EQUALS</code>	208
15. <code>public static final int RO_NOT_IN</code>	208
16. <code>public static final int RO_NOT_MATCHES</code>	208

Field detail

1)

Field	<code>public static final int RO_BETWEEN</code>
Description	Constant representing a BETWEEN operator.
Value	11

2)

Field	<code>public static final int RO_EQUALS</code>
Description	Constant representing the EQUALS operator.
Value	0

3)

Field	<code>public static final int RO_EXISTS</code>
Description	Constant representing the operation to check whether a value exists.
Value	10

4)

Field	<code>public static final int RO_GREATER_THAN</code>
Description	Constant representing the GREATER THAN operator.
Value	4

5)

Field	<code>public static final int RO_GREATER_THAN_OR_EQUAL</code>
Description	Constant representing the GREATER THAN OR EQUALS operator.
Value	5

6)

Field	<code>public static final int RO_IN</code>
Description	Constant representing the operation to check whether a value is in a list.
Value	6

7)

Field	<code>public static final int RO_IS_NOT_NULL</code>
Description	Constant representing the operation to check whether a value is not null.
Value	14

8)

Field	<code>public static final int RO_IS_NULL</code>
Description	Constant representing the operation to check whether a value is null.
Value	13

9)

Field	<code>public static final int RO_LESS_THAN</code>
Description	Constant representing the LESS THAN operator.
Value	2

10)

Field	<code>public static final int RO_LESS_THAN_OR_EQUAL</code>
Description	Constant representing the LESS THAN OR EQUALS operator.
Value	3

11)

Field	<code>public static final int RO_MATCHES</code>
Description	Constant representing the MATCHES operator.
Value	8

12)

Field	<code>public static final int RO_NONE</code>
Description	Constant representing that no operator has been defined.
Value	15

13)

Field	<code>public static final int RO_NOT_BETWEEN</code>
Description	Constant representing the operation to check whether a value is not between two other values.
Value	12

14)

Field	<code>public static final int RO_NOT_EQUALS</code>
Description	Constant representing the NOT EQUALS operator.
Value	1

15)

Field	<code>public static final int RO_NOT_IN</code>
Description	Constant representing the operation to check whether a value is not in a list.
Value	7

16)

Field	<code>public static final int RO_NOT_MATCHES</code>
Description	Constant representing the NOT MATCHES operator.
Value	9

Interface IRivRecordListener

(com.riversoft.riv_web.vertigo)

Description

This interface defines the methods that an object must implement to "listen" for the arrival of records from the transport layer. When data from Rendezvous is processed through the transport layer, registered IRivRecordListeners are notified by calling the `rrIRivRecordsReceived(CRivRecord[] rivRecs)` method.

Hierarchy

public interface *IRivRecordListener*

Method summary

Method	Reference Page
1. <code>public void rrlRivRecordsReceived(CRivRecord[] rivRecs)</code>	209
2. <code>public void rrlTimeOutReceived()</code>	209

Method detail

1)

Method `public void rrlRivRecordsReceived(CRivRecord[] rivRecs)`

Description Invoked when a list of CRivRecords is processed from the transport layer.

Parameters `rivRecs`—the array of CRivRecords which has been received from the transport layer.

2)

Method `public void rrlTimeOutReceived()`

Description Invoked when the time limit has expired for records to be received through the transport layer from Rendezvous.

Interface IRivSubjects

(com.riversoft.riv_web.vertigo)

Description

This interface defines the basic subject definitions used by all engines and clients in MWFM applications. Each service, such as `riv_class` or `riv_model`, has a set of subject definitions to be used depending on the type of request being sent, such as a query or request for notification. A query is a one-off request, whereas a request for notification is ongoing.

Hierarchy

public interface *IRivSubjects*

Field summary

Field	Reference Page
1. public static final String <i>RIV_AUTH_DATA_STORE_SUBJ</i>	211
2. public static final String <i>RIV_AUTH_INTERNAL_SUBJ</i>	212
3. public static final String <i>RIV_AUTH_NOTIFY_SUBJ</i>	212
4. public static final String <i>RIV_AUTH_QUERY_SUBJ</i>	212
5. public static final String <i>RIV_CLASS_DATA_STORE_SUBJ</i>	212
6. public static final String <i>RIV_CLASS_INTERNAL_SUBJ</i>	212
7. public static final String <i>RIV_CLASS_NOTIFY_SUBJ</i>	213
8. public static final String <i>RIV_CLASS_QUERY_SUBJ</i>	213
9. public static final String <i>RIV_CTRL_NOTIFY_SUBJ</i>	213
10. public static final String <i>RIV_CTRL_QUERY_SUBJ</i>	213
11. public static final String <i>RIV_DISCO_NOTIFY_SUBJ</i>	213
12. public static final String <i>RIV_DISCO_QUERY_SUBJ</i>	214

Field	Reference Page
13. <code>public static final String RIV_EVENT_DATA_STORE_SUBJ</code>	214
14. <code>public static final String RIV_EVENT_INTERNAL_SUBJ</code>	214
15. <code>public static final String RIV_EVENT_NOTIFY_SUBJ</code>	214
16. <code>public static final String RIV_EVENT_QUERY_SUBJ</code>	214
17. <code>public static final String RIV_MODEL_DATA_STORE_SUBJ</code>	215
18. <code>public static final String RIV_MODEL_INTERNAL_SUBJ</code>	215
19. <code>public static final String RIV_MODEL_NOTIFY_SUBJ</code>	215
20. <code>public static final String RIV_MODEL_QUERY_SUBJ</code>	215
21. <code>public static final String RIV_MONITOR_INTERNAL_SUBJ</code>	215
22. <code>public static final String RIV_MONITOR_NOTIFY_SUBJ</code>	216
23. <code>public static final String RIV_MONITOR_QUERY_SUBJ</code>	216
24. <code>public static final String RIV_MONITORAGENT_QUERY_SUBJ</code>	216
25. <code>public static final String RIV_MONITORAGENT_TOPODONE_SUBJ</code>	216
26. <code>public static final String RIV_REPOSIT_NOTIFY_SUBJ</code>	216
27. <code>public static final String RIV_REPOSIT_QUERY_SUBJ</code>	217

Field detail

1)

Field	<code>public static final String RIV_AUTH_DATA_STORE_SUBJ</code>
Description	Not yet in use.
Value	"RIVERSOFT.3.0.AUTH.DATA"

2)

Field `public static final String RIV_AUTH_INTERNAL_SUBJ`

Description Not yet in use.

Value `"RIVERSOFT.3.0.AUTH.INTRNL"`

3)

Field `public static final String RIV_AUTH_NOTIFY_SUBJ`

Description Notify subject for `riv_auth`.

Value `"RIVERSOFT.3.0.AUTH.NOTIFY"`

4)

Field `public static final String RIV_AUTH_QUERY_SUBJ`

Description Query subject for `riv_auth`.

Value `"RIVERSOFT.3.0.AUTH.QUERY"`

5)

Field `public static final String RIV_CLASS_DATA_STORE_SUBJ`

Description Not yet in use.

Value `"RIVERSOFT.3.0.CLASS.DATA"`

6)

Field `public static final String RIV_CLASS_INTERNAL_SUBJ`

Description Not yet in use.

Value `"RIVERSOFT.3.0.CLASS.INTRNL"`

7)

Field	<code>public static final String RIV_CLASS_NOTIFY_SUBJ</code>
Description	Notify subject definition for <code>riv_class</code> .
Value	"RIVERSOFT.3.0.CLASS.NOTIFY"

8)

Field	<code>public static final String RIV_CLASS_QUERY_SUBJ</code>
Description	Query subject definition for <code>riv_class</code> .
Value	"RIVERSOFT.3.0.CLASS.QUERY"

9)

Field	<code>public static final String RIV_CTRL_NOTIFY_SUBJ</code>
Description	Notify subject for <code>riv_ctrl</code> .
Value	"RIVERSOFT.3.0.CTRL.NOTIFY"

10)

Field	<code>public static final String RIV_CTRL_QUERY_SUBJ</code>
Description	Query subject for <code>riv_ctrl</code> .
Value	"RIVERSOFT.3.0.CTRL.QUERY"

11)

Field	<code>public static final String RIV_DISCO_NOTIFY_SUBJ</code>
Description	Notify subject for <code>riv_disco</code> .
Value	"RIVERSOFT.3.0.DISCO.NOTIFY"

12)

Field	<code>public static final String RIV_DISCO_QUERY_SUBJ</code>
Description	Query subject for <code>riv_disco</code> .
Value	<code>"RIVERSOFT.3.0.DISCO.QUERY"</code>

13)

Field	<code>public static final String RIV_EVENT_DATA_STORE_SUBJ</code>
Description	Not yet in use.
Value	<code>"RIVERSOFT.3.0.EVENT.DATA"</code>

14)

Field	<code>public static final String RIV_EVENT_INTERNAL_SUBJ</code>
Description	Not yet in use.
Value	<code>"RIVERSOFT.3.0.EVENT.INTRNL"</code>

15)

Field	<code>public static final String RIV_EVENT_NOTIFY_SUBJ</code>
Description	Notify subject definition for <code>riv_f_amos</code> .
Value	<code>"RIVERSOFT.3.0.EVENT.NOTIFY"</code>

16)

Field	<code>public static final String RIV_EVENT_QUERY_SUBJ</code>
Description	Query subject definition for <code>riv_f_amos</code> .
Value	<code>"RIVERSOFT.3.0.EVENT.QUERY"</code>

17)

Field	<code>public static final String RIV_MODEL_DATA_STORE_SUBJ</code>
Description	Not yet in use.
Value	"RIVERSOFT.3.0.MODEL.DATA"

18)

Field	<code>public static final String RIV_MODEL_INTERNAL_SUBJ</code>
Description	Not yet in use.
Value	"RIVERSOFT.3.0.MODEL.INTRNL"

19)

Field	<code>public static final String RIV_MODEL_NOTIFY_SUBJ</code>
Description	Notify subject definition for <code>riv_model</code> .
Value	"RIVERSOFT.3.0.MODEL.NOTIFY"

20)

Field	<code>public static final String RIV_MODEL_QUERY_SUBJ</code>
Description	Query subject definition for <code>riv_model</code> .
Value	"RIVERSOFT.3.0.MODEL.QUERY"

21)

Field	<code>public static final String RIV_MONITOR_INTERNAL_SUBJ</code>
Description	Agent Reply subject for <code>riv_monitor</code> .
Value	"RIVERSOFT.3.0.MONITOR.REPLYFROMAGENT"

22)

Field `public static final String RIV_MONITOR_NOTIFY_SUBJ`

Description Notify subject for `riv_monitor`.

Value `"RIVERSOFT.3.0.MONITOR.NOTIFY"`

23)

Field `public static final String RIV_MONITOR_QUERY_SUBJ`

Description Query subject for `riv_monitor`.

Value `"RIVERSOFT.3.0.MONITOR.QUERY"`

24)

Field `public static final String RIV_MONITORAGENT_QUERY_SUBJ`

Description Query subject for `riv_m_agent`.

Value `"RIVERSOFT.3.0.MONITORAGENT.QUERY"`

25)

Field `public static final String RIV_MONITORAGENT_TOPODONE_SUBJ`

Description Topo Done subject for `riv_m_agent`.

Value `"RIVERSOFT.3.0.MONITORAGENT.TOPODONE"`

26)

Field `public static final String RIV_REPOSIT_NOTIFY_SUBJ`

Description Notify subject for `riv_store`.

Value `"RIVERSOFT.3.0.STORE.NOTIFY"`

27)

Field `public static final String RIV_REPOSIT_QUERY_SUBJ`Description Query subject for `riv_store`.Value `"RIVERSOFT.3.0.STORE.QUERY"`

Interface IRivTimerCallback

(com.riversoft.riv_web.vertigo)

Description

`IRivTimerCallback` defines the interface for handler classes of timer event activities in the client.

Hierarchy

public interface *IRivTimerCallback*

Method summary

Method	Reference Page
1. <code>public void rtcTimer()</code>	217

Method detail

1)

Method `public void rtcTimer()`

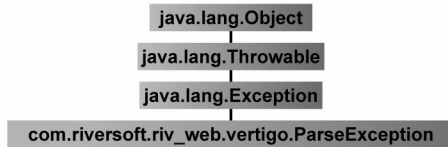
Description The `rtcTimer()` method will process each timer event as it occurs. This is an asynchronous callback function and should return promptly. If this method must perform lengthy computations it is recommended that this method should spawn a separate thread.

Class ParseException

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-25 Hierarchy of class ParseException



```
public class ParseException
extends java.lang.Exception
```

Description

This exception is thrown when parse errors are encountered. You can explicitly create objects of this exception type by calling the method `generateParseException` in the generated parser.

ParseException

This class is only used by the parsers— `CRivFilterParser` and `CRivMonitorFilterParser`. Application-level code should never need to use this class directly.

See Also

`RivFilter`, Serialised Form

Field summary

Field	Reference Page
1. public Token <i>currentToken</i>	219
2. protected String <i>eol</i>	220
3. public int[][] <i>expectedTokenSequences</i>	220
4. protected boolean <i>specialConstructor</i>	220
5. public String[] <i>tokenImage</i>	220

Constructor summary

Constructor	Reference Page
1. public <i>ParseException</i> ()	220
2. public <i>ParseException</i> (String message)	221
3. public <i>ParseException</i> (Token currentTokenVal, int[][] expectedTokenSequencesVal, String[] tokenImageVal)	221

Method summary

Method	Reference Page
1. protected String <i>add_escapes</i> (String str)	222
2. public String <i>getMessage</i> ()	222

Field detail

1)

Field `public Token currentToken`

Description This is the last token that has been consumed successfully. If this object has been created due to a parse error, the token following this token will therefore be the first error token.

2)

Field `protected String eol`

Description The end of line `String` for this machine.

3)

Field `public int[][] expectedTokenSequences`

Description Each entry in this array is an array of `integers`. Each array of `integers` represents a sequence of tokens (by their ordinal values) that is expected at this point of the parse.

4)

Field `protected boolean specialConstructor`

Description This variable determines which constructor was used to create this object and thereby affects the semantics of the `"getMessage"` method (see below).

5)

Field `public String[] tokenImage`

Description This is a reference to the `"tokenImage"` array of the generated parser within which the parse error occurred. This array is defined in the generated `...Constants` interface, i.e., `IRivConstants` or `RivFilterConstants` interfaces.

Constructor detail

1)

Constructor `public ParseException()`

Description The following constructors are for use by developers for multiple purposes. Constructing the exception in this manner makes the exception behave in the normal way, i.e., as documented in the class `"Throwable"`. The fields `"currentToken"`, `"expectedTokenSequences"`, and `"tokenImage"` do not contain relevant information.

2)

Constructor	<code>public ParseException(String message)</code>
Description	Construct a <code>ParseException</code> with the specified detail message that will describe this particular exception.
Parameters	<code>message</code> —the detail message.

3)

Constructor	<code>public ParseException(Token currentTokenVal, int[] [] expectedTokenSequencesVal, String[] tokenImageVal)</code>
Description	This constructor is used by the method "generateParseException" in the generated parser. Calling this constructor generates a new object of this type with the fields "currentToken", "expectedTokenSequences", and "tokenImage" set. The boolean flag "specialConstructor" is also set to <code>true</code> to indicate that this constructor was used to create this object. This constructor calls its super class with the empty <code>String</code> to force the "toString" method of parent class "Throwable" to print the error message in the form: <code>ParseException</code> .
Parameters	<code>currentTokenVal</code> —the current token which has caused this exception. <code>expectedTokenSequencesVal</code> —the sequence of token values that were expected. <code>tokenImageVal</code> —the token image value.

Method detail

1)

Method `protected String add_escapes(String str)`

Description Used to convert raw characters to their escaped version when these raw versions cannot be used as part of an ASCII String literal.

Parameters `str`—the String to convert to the escaped version.

2)

Method `public String getMessage()`

Description This method has the standard behaviour when this object has been created using the standard constructors. Otherwise, it uses "currentToken" and "expectedTokenSequences" to generate a parse error message and returns it. If this object has been created due to a parse error, and you do not catch it (it gets thrown from the parser), then this method is called during the printing of the final stack trace, and hence the correct error message gets displayed.

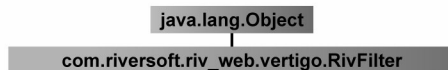
Overrides `getMessage` in class `java.lang.Throwable`

Class RivFilter

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Figure 4-26 Hierarchy of class RivFilter



```

public class RivFilter
  extends java.lang.Object
  implements IRivOper, RivFilterConstants
  
```

Description

This class defines the parser for filter objects, allowing a text `String` in the appropriate format to be parsed to a MWFM filter object, `CRivFilter`.

This parser is commonly used to process filter objects which arrive from MWFM data engines via OQL requests contained in `CRivAtom` objects of type `IRivDataType.RDT_STRING`.



Note

`RivFilter`—This class should never be used directly, but should always be accessed through the class `CRivFilterParser`, by calling its `rpfParse(String text)` method, which will return a `CRivFilter` that has been parsed from the given text `String`.

See Also

`CRivFilter`, `CRivFilterParser`, `IRivOper`

Field summary

Field

1. `public Token jj_nt`
2. `public boolean lookingAhead`
3. `public Token token`
4. `public RivFilterTokenManager token_source`

Constructor summary

Constructor	Reference Page
1. <code>public RivFilter(java.io.InputStream stream)</code>	225
2. <code>public RivFilter(java.io.Reader stream)</code>	225
3. <code>public RivFilter(RivFilterTokenManager tm)</code>	225

Method summary

Method

1. `public final void disable_tracing()`
2. `public final void enable_tracing()`
3. `public final ParseException generateParseException()`
4. `public final void getArray()`
5. `public final CRivAtom getAtom()`

Method
6. public final java.util.Vector <i>getAtomList</i> ()
7. public final CRivExpr <i>getBetweenPredicate</i> ()
8. public final CRivDbEntity <i>getColumnRef</i> ()
9. public final int <i>getComparison</i> ()
10. public final CRivQueryAtom <i>getComparisonPredicate</i> (CRivExpr lhsExpr)
11. public final CRivFilter <i>getCondition</i> ()
12. public final int <i>getDataType</i> ()
13. public final CRivEvalClause <i>getEvalRef</i> ()
14. public final CRivFilter <i>getFilter</i> ()
15. public final CRivExpr <i>getInPredicate</i> ()
16. public final CRivExpr <i>getLikePredicate</i> ()
17. public final void <i>getList</i> ()
18. public final CRivAtom <i>getListWhole</i> ()
19. public final Token <i>getNextToken</i> ()
20. public final void <i>getObject</i> ()
21. public final CRivAtom <i>getObjectWhole</i> ()
22. public final CRivQueryAtom <i>getPredicate</i> ()
23. public final CRivExpr <i>getScalarExp</i> ()
24. public final Token <i>getToken</i> (int index)
25. public final CRivVarBind <i>getVarBind</i> ()
26. public final CRivVarBindList <i>getVarBindList</i> ()
27. public final CRivAtom <i>optEscape</i> ()
28. public final CRivFilter <i>parseInput</i> ()
29. public CRivFilter <i>parseInput</i> (java.io.InputStream fileStream)
30. public CRivFilter <i>parseInput</i> (java.io.Reader reader)
31. public void <i>ReInit</i> (java.io.InputStream stream)
32. public void <i>ReInit</i> (java.io.Reader stream)
33. public void <i>ReInit</i> (RivFilterTokenManager tm)
34. public final CRivQueryAtom <i>testForNull</i> (CRivExpr lhsExpr)

Constructor Detail

1)

Constructor	<code>public RivFilter(java.io.InputStream stream)</code>
Description	Construct a new <code>FilterParser</code> to parse the specified <code>InputStream</code> .
Parameters	<code>stream</code> —the input stream to parse to a filter.

2)

Constructor	<code>public RivFilter(java.io.Reader stream)</code>
Description	Construct a new <code>FilterParser</code> to parse the specified character input stream.
Parameters	<code>stream</code> —the character input stream to parse to a filter.

3)

Constructor	<code>public RivFilter(RivFilterTokenManager tm)</code>
Description	Construct a new <code>FilterParser</code> for the specified <code>RivFilterTokenManager</code> .
Parameters	<code>tm</code> —the Token Manager for this parser.

Interface RivFilterConstants

`(com.riversoft.riv_web.vertigo)`

Description

This interface defines the tokens and keywords used by the Filter parser, `RivFilter`. Since application-level code should always access this parser through the class `CRivFilterParser`, the constants in this interface do not need to be used directly.

All Known Implementing Classes:

`RivFilter`, `RivFilterTokenManager`.

Field summary

Field
1. public static final int DEFAULT
2. public static final int <i>DIGIT</i>
3. public static final int <i>EOF</i>
4. public static final int <i>RIV_FLT_AND</i>
5. public static final int <i>RIV_FLT_APPROXNUM</i>
6. public static final int <i>RIV_FLT_ARRAY</i>
7. public static final int <i>RIV_FLT_BETWEEN</i>
8. public static final int <i>RIV_FLT_CHARACTER</i>
9. public static final int <i>RIV_FLT_DATA</i>
10. public static final int <i>RIV_FLT_EQ</i>
11. public static final int <i>RIV_FLT_ESCAPE</i>
12. public static final int <i>RIV_FLT_EVAL</i>
13. public static final int <i>RIV_FLT_FLOAT</i>
14. public static final int <i>RIV_FLT_GT</i>
15. public static final int <i>RIV_FLT_GTE</i>
16. public static final int <i>RIV_FLT_HAVING</i>
17. public static final int <i>RIV_FLT_IDENT</i>
18. public static final int <i>RIV_FLT_IN</i>
19. public static final int <i>RIV_FLT_INTEGER</i>
20. public static final int <i>RIV_FLT_INTNUM</i>
21. public static final int <i>RIV_FLT_IPADDRESS</i>
22. public static final int <i>RIV_FLT_IS</i>
23. public static final int <i>RIV_FLT_LIKE</i>
24. public static final int <i>RIV_FLT_LIST</i>
25. public static final int <i>RIV_FLT_LOCATOR</i>
26. public static final int <i>RIV_FLT_LONG</i>
27. public static final int <i>RIV_FLT_LONGLONG</i>
28. public static final int <i>RIV_FLT_LT</i>
29. public static final int <i>RIV_FLT_LTE</i>
30. public static final int <i>RIV_FLT_NEQ</i>
31. public static final int <i>RIV_FLT_NOT</i>
32. public static final int <i>RIV_FLT_NULLX</i>
33. public static final int <i>RIV_FLT_OBJECT</i>
34. public static final int <i>RIV_FLT_OR</i>
35. public static final int <i>RIV_FLT_SMALLINT</i>

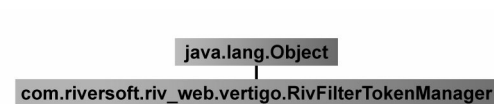
Field
36. public static final int <i>RIV_FLT_STRING</i>
37. public static final int <i>RIV_FLT_TEXT</i>
38. public static final int <i>RIV_FLT_TIME</i>
39. public static final int <i>RIV_FLT_TYPE</i>
40. public static final int <i>RIV_FLT_VARCHAR</i>
41. public static final int <i>RIV_FLT_WHERE</i>
42. public static final String[] <i>tokenImage</i>

Class RivFilterTokenManager

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-27 Hierarchy of class RivFilterTokenManager



```

public class RivFilterTokenManager
extends java.lang.Object
implements RivFilterConstants
  
```

Description

This class manages the tokens and keywords used by the Filter parser, *RivFilter*. Since application-level code should always access this parser through the class *CRivFilterParser*, this class need never be called directly.

RivFilterTokenManager

This class is only used by the parsers— *CRivFilterParser* and *CRivMonitorFilterParser*. Application-level code should never need to use this class directly. The summaries are included for reference only and thus no descriptions are included.

See Also

RivFilter, *CRivFilterParser*

Field summary

Field

1. protected char *curChar*
 2. public static final String[] *jjstrLiteralImages*
 3. public static final String[] *lexStateNames*
-

Constructor summary

Constructor

1. public *RivFilterTokenManager*(ASCII_CharStream stream)
 2. public *RivFilterTokenManager*(ASCII_CharStream stream, int lexState)
-

Method summary

Method

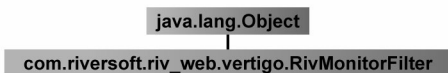
1. public final Token getNextToken()
 2. public void *ReInit*(ASCII_CharStream stream)
 3. public void *ReInit*(ASCII_CharStream stream, int lexState)
 4. public void *SwitchTo*(int lexState)
-

Class RivMonitorFilter

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-28 Hierarchy of class RivMonitorFilter



```

public class RivMonitorFilter
extends java.lang.Object
implements IRivOper, RivMonitorFilterConstants
  
```

Description

This class defines the parser for filter objects, allowing a text `String` in the appropriate format to be parsed to a MWFM filter object, `CRivFilter`.

This parser differs from `RivFilter` by recognising name fields in filters which include the style of addressing "Name[.x]+", i.e., a name, followed by one or more ".x" where x is any number, e.g., "ifOperStatus.1". This style of addressing would not be parsed successfully by `RivFilter`. This is commonly encountered in defining Threshold filters in Polling Agents, particularly SNMP agents, used by `riv_monitor`.

This parser is commonly used to process filter objects which arrive from MWFM data engines via OQL requests contained in `CRivAtom` objects of type `IRivDataType.RDT_STRING`.



Note

RivMonitorFilter—This class should never be used directly, but should always be accessed through the class `CRivMonitorFilterParser`, by calling its `rfpParse(String text)` method, which will return a `CRivFilter` that has been parsed from the given text `String`.

See Also

`CRivFilter`, `CRivMonitorFilterParser`, `IRivOper`, `RivFilter`

Field summary

Field
1. <code>public Token jj_nt</code>
2. <code>public boolean lookingAhead</code>
3. <code>public Token token</code>
4. <code>public RivMonitorFilterTokenManager token_source</code>

Constructor summary

Constructor	Reference Page
1. <code>public RivMonitorFilter(java.io.InputStream stream)</code>	231
2. <code>public RivMonitorFilter(java.io.Reader stream)</code>	231
3. <code>public RivMonitorFilter(RivMonitorFilterTokenManager tm)</code>	231

Method summary

Method
1. public final void <i>disable_tracing</i> ()
2. public final void <i>enable_tracing</i> ()
3. public final ParseException <i>generateParseException</i> ()
4. public final void <i>getArray</i> ()
5. public final CRivAtom <i>getAtom</i> ()
6. public final java.util.Vector <i>getAtomList</i> ()
7. public final CRivExpr <i>getBetweenPredicate</i> ()
8. public final CRivDbEntity <i>getColumnRef</i> ()
9. public final int <i>getComparison</i> ()
10. public final CRivQueryAtom <i>getComparisonPredicate</i> (CRivExpr lhsExpr)
11. public final CRivFilter <i>getCondition</i> ()
12. public final int <i>getDataType</i> ()
13. public final CRivEvalClause <i>getEvalRef</i> ()
14. public final CRivFilter <i>getFilter</i> ()
15. public final CRivExpr <i>getInPredicate</i> ()
16. public final CRivExpr <i>getLikePredicate</i> ()
17. public final void <i>getList</i> ()
18. public final CRivAtom <i>getListWhole</i> ()
19. public final Token <i>getNextToken</i> ()
20. public final void <i>getObject</i> ()
21. public final CRivAtom <i>getObjectWhole</i> ()
22. public final CRivQueryAtom <i>getPredicate</i> ()
23. public final CRivExpr <i>getScalarExp</i> ()
24. public final Token <i>getToken</i> (int index)
25. public final CRivVarBind <i>getVarBind</i> ()
26. public final CRivVarBindList <i>getVarBindList</i> ()
27. public final CRivAtom <i>optEscape</i> ()
28. public final CRivFilter <i>parseInput</i> ()
29. public CRivFilter <i>parseInput</i> (java.io.InputStream fileStream)
30. public CRivFilter <i>parseInput</i> (java.io.Reader reader)
31. public void <i>ReInit</i> (java.io.InputStream stream)
32. public void <i>ReInit</i> (java.io.Reader stream)

Method

33. `public void ReInit(RivMonitorFilterTokenManager tm)`
-
34. `public final CRivQueryAtom testForNull(CRivExpr lhsExpr)`
-

Constructor detail

1)

Constructor `public RivMonitorFilter(java.io.InputStream stream)`

Description Construct a new `FilterParser` with which to parse the specified `InputStream`.

Parameters `stream`—the input stream to parse for a filter.

2)

Constructor `public RivMonitorFilter(java.io.Reader stream)`

Description Construct a new `FilterParser` with which to parse the specified character input stream.

Parameters `stream`—the character input stream to parse for a filter.

3)

Constructo `public RivMonitorFilter (RivMonitorFilterTokenManager tm)`
r

Description Construct a new `FilterParser` for the specified `RivMonitorFilterTokenManager`.

Parameters `tm`—the Token Manager for this parser.

Interface RivMonitorFilterConstants

(com.riversoft.riv_web.vertigo)

Description

This interface defines the tokens and keywords used by the Filter parser, `RivMonitorFilter`, which is commonly encountered in parsing threshold filters in Polling Agents used by `riv_monitor`. Since application-level code should always access this parser through the class `CRivMonitorFilterParser`, the constants in this interface should not be used directly.

RivMonitorFilterConstants

This class is only used by the parsers `CRivFilterParser` and `CRivMonitorFilterParser`. Application-level code should never need to use this class directly. The field summaries are included for reference only and thus no descriptions are included.

Hierarchy

```
public interface RivMonitorFilterConstants
```

All Known Implementing Classes

`RivMonitorFilter`, `RivMonitorFilterTokenManager`

See Also

`RivMonitorFilter`, `CRivMonitorFilterParser`

Field summary

Field
1. public static final int DEFAULT
2. public static final int DIGIT
3. public static final int EOF
4. public static final int RIV_FLT_AND
5. public static final int RIV_FLT_APPROXNUM
6. public static final int RIV_FLT_ARRAY
7. public static final int RIV_FLT_BETWEEN
8. public static final int RIV_FLT_CHARACTER
9. public static final int RIV_FLT_DATA
10. public static final int RIV_FLT_EQ
11. public static final int RIV_FLT_ESCAPE

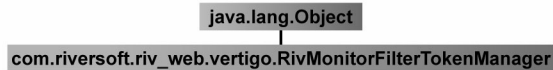
Field
12. public static final int <i>RIV_FLT_EVAL</i>
13. public static final int <i>RIV_FLT_FLOAT</i>
14. public static final int <i>RIV_FLT_GT</i>
15. public static final int <i>RIV_FLT_GTE</i>
16. public static final int <i>RIV_FLT_HAVING</i>
17. public static final int <i>RIV_FLT_IDENT</i>
18. public static final int <i>RIV_FLT_IN</i>
19. public static final int <i>RIV_FLT_INTEGER</i>
20. public static final int <i>RIV_FLT_INTNUM</i>
21. public static final int <i>RIV_FLT_IPADDRESS</i>
22. public static final int <i>RIV_FLT_IS</i>
23. public static final int <i>RIV_FLT_LIKE</i>
24. public static final int <i>RIV_FLT_LIST</i>
25. public static final int <i>RIV_FLT_LOCATOR</i>
26. public static final int <i>RIV_FLT_LONG</i>
27. public static final int <i>RIV_FLT_LONGLONG</i>
28. public static final int <i>RIV_FLT_LT</i>
29. public static final int <i>RIV_FLT_LTE</i>
30. public static final int <i>RIV_FLT_NEQ</i>
31. public static final int <i>RIV_FLT_NOT</i>
32. public static final int <i>RIV_FLT_NULLX</i>
33. public static final int <i>RIV_FLT_OBJECT</i>
34. public static final int <i>RIV_FLT_OR</i>
35. public static final int <i>RIV_FLT_SMALLINT</i>
36. public static final int <i>RIV_FLT_STRING</i>
37. public static final int <i>RIV_FLT_TEXT</i>
38. public static final int <i>RIV_FLT_TIME</i>
39. public static final int <i>RIV_FLT_TYPE</i>
40. public static final int <i>RIV_FLT_VARCHAR</i>
41. public static final int <i>RIV_FLT_WHERE</i>
42. public static final String[] <i>tokenImage</i>

Class RivMonitorFilterTokenManager

(com.riversoft.riv_web.vertigo)

Hierarchy

Figure 4-29 Hierarchy of class RivMonitorFilterTokenManager



```

public class RivMonitorFilterTokenManager
  extends java.lang.Object
  implements RivMonitorFilterConstants
  
```

Description

This class manages the tokens and keywords used by the Filter parser, `RivMonitorFilter`, which is commonly encountered in parsing threshold filters in Polling Agents used by `riv_monitor`. Since application-level code should always access this parser through the class `CRivMonitorFilterParser`, this class need never be called directly.

RivMonitorFilterTokenManager

This class is only used by the parsers—`CRivFilterParser` and `CRivMonitorFilterParser`. Application-level code should never need to use this class directly. The summaries are included for reference only and thus no descriptions are included.

See Also

`RivMonitorFilter`, `CRivMonitorFilterParser`

Field summary

Field

1. `protected char curChar`
 2. `public static final String[] jjstrLiteralImages`
 3. `public static final String[] lexStateNames`
-

Constructor summary

Constructor

1. `public RivMonitorFilterTokenManager(ASCII_CharStream stream)`
 2. `public RivMonitorFilterTokenManager(ASCII_CharStream stream, int lexState)`
-

Method summary

Method

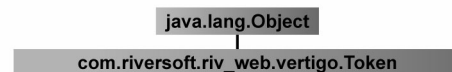
1. `public final Token getNextToken()`
 2. `public void ReInit(ASCII_CharStream stream)`
 3. `public void ReInit(ASCII_CharStream stream, int lexState)`
 4. `public void SwitchTo(int lexState)`
-

Class Token

(`com.riversoft.riv_web.vertigo`)

Hierarchy

Figure 4-30 Hierarchy of class *Token*



```
public class Token
extends java.lang.Object
```

Description

Describes the input token stream.

Token

This class is only used by the parsers—`CRivFilterParser` and `CRivMonitorFilterParser`. Application-level code should never need to use this class directly.

See Also

`RivFilter`

Field summary

Field	Reference Page
1. <code>public int beginColumn</code>	237
2. <code>public int beginLine</code>	237
3. <code>public int endColumn</code>	237
4. <code>public int endLine</code>	237
5. <code>public String image</code>	237
6. <code>public int kind</code>	238
7. <code>public Token next</code>	238
8. <code>public Token specialToken</code>	238

Constructor summary

Constructor	Reference Page
1. <code>public Token()</code>	238

Method summary

Method	Reference Page
1. <code>public static final Token newToken(int ofKind)</code>	239
2. <code>public final String toString()</code>	239

Field detail

1)

Field `public int beginColumn`

Description `beginLine` and `beginColumn` describe the position of the first character of this token; `endLine` and `endColumn` describe the position of the last character of this token.

2)

Field `public int beginLine`

Description `beginLine` and `beginColumn` describe the position of the first character of this token; `endLine` and `endColumn` describe the position of the last character of this token.

3)

Field `public int endColumn`

Description `beginLine` and `beginColumn` describe the position of the first character of this token; `endLine` and `endColumn` describe the position of the last character of this token.

4)

Field `public int endLine`

Description `beginLine` and `beginColumn` describe the position of the first character of this token; `endLine` and `endColumn` describe the position of the last character of this token.

5)

Field `public String image`

Description The `String` image of the token.

6)

Field `public int kind`

Description An integer that describes the kind of 'this' token.

7)

Field `public Token next`

Description A reference to the next regular (non-special) token from the input stream. If this is the last token from the input stream, or if the token manager has not read tokens beyond this one, this field is set to `null`. This is `true` only if this token is also a regular token. Otherwise, see below for a description of the contents of this field.

8)

Field `public Token specialToken`

Description This field is used to access special tokens that occur prior to this token, but after the immediately preceding regular (non-special) token. If there are no such special tokens, this field is set to `null`. When there are more than one such special tokens, this field refers to the last of these special tokens, which in turn refers to the previous special token through its `specialToken` field, and so on until the first special token (whose `specialToken` field is `null`). The next fields of special tokens refer to other special tokens that immediately follow it (without an intervening regular token). If there is no such token, this field is `null`.

Constructor detail

Constructor `public Token()`

Description Create a `Token` object.

Method detail

1)

Method `public static final Token newToken(int ofKind)`

Description Returns a new `Token` object by default. However, if you want, you can create and return subclass objects based on the value of `ofKind`. Simply add the cases to the switch for all those special cases. For example, if you have a subclass of `Token` called `IDToken` that you want to create and if `ofKind` is `ID`, simply add something like `: case MyParserConstants.ID : return new IDToken ();` to the following switch statement. Then you can cast `matchedToken` variable to the appropriate type and use it in your lexical actions.

Parameters `ofKind`—the kind of `Token`.

2)

Method `public final String toString()`

Description Returns the `String` representation of this `Token`.

Overrides `toString` in class `java.lang.Object`

Class TokenMgrError

`(com.riversoft.riv_web.vertigo)`

Hierarchy

Figure 4-31 Hierarchy of class `TokenMgrError`



```
public class TokenMgrError
extends javac.lang.Error
```

Description

This class is used by parsers, such as `RivFilter` and application-level code should never need to use this class.

TokenMgrError

This class is only used by the parsers—`CRivFilterParser` and `CRivMonitorFilterParser`. Application-level code should never need to use this class directly. The constructor summaries are included for reference only and thus have no descriptions.

See Also

`RivFilter`, Serialised Form

Constructor summary

Constructor

1. `public TokenMgrError()`
2. `public TokenMgrError(boolean EOFSeen, int lexState, int errorLine, int errorColumn, String errorAfter, char curChar, int reason)`
3. `public TokenMgrError(String message, int reason)`

Method summary

Method	Reference Page
1. <code>protected static final String <i>addEscapes</i>(String str)</code>	240

Method detail

1)

Method `protected static final String addEscapes(String str)`

Description Replaces unprintable characters by their escaped (or unicode escaped) equivalents in the given `String`.

Parameters `str`—the `String` to replace with its escaped equivalent.

Class/Interface index

Class/Interface	Index Page
1. ASCII_CharStream	242
2. CRivASNIAddress	243
3. CRivAtom	243
4. CRivClient	244
5. CRivClientHelper	244
6. CRivClientInterface	245
7. CRivClientInterface.RCIRRecordListener	245
8. CRivClientInterface.RCITimerCallback	246
9. CRivDbEntity	246
10. CRivDummyCallback	246
11. CRivEvalClause	246
12. CRivException	247
13. CRivException.RivErrRec	247
14. CRivExpr	248
15. CRivFilter	248
16. CRivFilterParser	248
17. CRivHashVector	249
18. CRivMonitorFilterParser	249
19. CRivQueryAtom	249
20. CRivRecord	250
21. CRivROMPer	250
22. CRivRvDataHandler	251
23. CRivTransport	251
24. CRivVarBind	252
25. CRivVarBindList	252
26. IRivAlgebraic	253
27. IRivConstants	254
28. IRivDataType	255
29. IRivNodeType	255
30. IRivOper	256
31. IRivRecordListener	256
32. IRivSubjects	256
33. IRivTimerCallback	257
34. ParseException	257
35. RivFilter	257

Class/Interface	Index Page
36. RivFilterConstants	258
37. RivFilterTokenManager	259
38. RivMonitorFilter	259
39. RivMonitorFilterConstants	260
40. RivMonitorFilterTokenManager	261
41. Token	261
42. TokenMgrError	261

Class *ASCII_CharStream*

Fields

```
public int bufpos
public static final boolean staticFlag
```

Constructors

```
public ASCII_CharStream(java.io.InputStream dstream, int startline, int
startcolumn)
public ASCII_CharStream(java.io.InputStream dstream, int startline, int
startcolumn, int buffersize)
public ASCII_CharStream(java.io.Reader dstream, int startline, int startcolumn)
public ASCII_CharStream(java.io.Reader dstream, int startline, int startcolumn,
int buffersize)
```

Public instance methods

```
public void adjustBeginLineColumn(int newLine, int newCol)
public final void backup(int amount)
public final char BeginToken()
public void Done()
public final int getBeginColumn()
public final int getBeginLine()
public final int getColumn()
public final int getEndColumn()
public final int getEndLine()
public final String GetImage()
public final int getLine()
public final char[] GetSuffix(int len)
public final char readChar()
public void ReInit(java.io.InputStream dstream, int startline, int startcolumn)
public void ReInit(java.io.InputStream dstream, int startline, int startcolumn,
int buffersize)
public void ReInit(java.io.Reader dstream, int startline, int startcolumn)
public void ReInit(java.io.Reader dstream, int startline, int startcolumn, int
buffersize)
```


Class *CRivASN1Address*

Field

```
public static int RIV_MAX_ADDR_DEPTH
```

Constructors

```
public CRivASN1Address()
public CRivASN1Address(CRivASN1Address arg)
public CRivASN1Address(CRivASN1Address parent, int child)
public CRivASN1Address(int firstNEntries, CRivASN1Address arg)
public CRivASN1Address(int relative, int[] full, int depth)
public CRivASN1Address(String printable)
```

Public instance methods

```
public int getRAAAddressAt(int position)
public int getRAADepth()
public int[] getRAAFullAddress()
public int getRAARelativeAddress()
public boolean isRAAMatchAddress(CRivASN1Address arg)
public boolean isRAAPartialMatch(CRivASN1Address arg)
public CRivASN1Address raaUnion(CRivASN1Address base, CRivASN1Address qual)
public String toString()
```

Class *CRivAtom*

Constructors

```
public CRivAtom()
public CRivAtom(byte[] value)
public CRivAtom(CRivASN1Address value)
public CRivAtom(CRivAtom atomToCopy)
public CRivAtom(CRivVarBindList value)
public CRivAtom(float value)
public CRivAtom(int value)
public CRivAtom(long value)
public CRivAtom(String value)
public CRivAtom(java.util.Vector value)
```

Public instance methods

```
public boolean equals(Object obj)
public CRivASN1Address getRAASN1Address()
public byte[] getRAByteArray()
public java.lang.Class getRADataClass()
public float getRAFloatValue()
public int getRAIntValue()
public long getRALongValue()
public CRivVarBindList getRAObject()
public CRivAtom getRASubVal(String name)
public int getRAType()
public java.util.Vector getRAVector()
public CRivAtom raDeepCopyListAtom()
public CRivAtom raDeepCopyObjectAtom()
public boolean raEvalRelop(int operator, CRivAtom atomToCompare)
public boolean raIsAtomInVector(CRivAtom entry)
public int raLex(CRivAtom comp)
```

```

public void setRAValue(CRivASN1Address value)
public void setRAValue(CRivVarBindList value)
public void setRAValue(float value)
public void setRAValue(int value)
public void setRAValue(String value)
public void setRAValue(java.util.Vector value)
public String toString()

```

Class *CRivClient*

Field

```
public static final long RIV_DEFAULT_LATENCY
```

Constructor

```
public CRivClient(long timeOut, String domain)
```

Public instance methods

```

public String getRCDomain()
public String getRCDomainSubj(String subjStub)
public long getRCTimeOut()
public boolean isRCValidOutboundType(Object data)
public COM.TIBCO.rv.RvTimer rcAddOnceOnlyTimer(long interval, COM.TIBCO.rv.RvTimerCallback handler)
public COM.TIBCO.rv.RvTimer rcAddRepeatTimer(long interval, COM.TIBCO.rv.RvTimerCallback handler)
public int rcAddService(String subject, COM.TIBCO.rv.RvTimerCallback updateHandler)
public int rcInitSession()
public int rcInitSession(String hostname)
public int rcInitSession(String hostname, int port)
public int rcInitSession(String service, String network, String daemon)
public boolean rcRemoveTimer(RvTimer timer)
public int rcSend(String subject, Object data)
public int rcSend(String subject, Object data, COM.TIBCO.rv.RvTimerCallback handler)
public int rcTerminateSession()
public int rcTimedSend(String subject, Object data, long timeout, COM.TIBCO.rv.RvTimerCallback handler)
public void setRCTimeOut(long timeout)

```

Class *CRivClientHelper*

Constructors

```
public CRivClientHelper(CRivClient client, String username, String password)
```

Methods

```

public CRivRecord[] getRCHAllEventRecords()
public CRivRecord[] getRCHAllModelRecords()
public boolean getRCHAuthPermission(String category, String action)
public CRivRecord[] getRCHAuthProfiles()
public CRivRecord[] getRCHAuthUsers()
public CRivRecord[] getRCHEventRecords(CRivFilter filter)
public CRivRecord[] getRCHEventRecords(CRivVarBindList vbList)
public CRivRecord[] getRCHInterfaces(CRivRecord modelRec)
public CRivRecord[] getRCHModelRecords(CRivFilter filter)
public CRivRecord[] getRCHModelRecords(CRivVarBindList vbList)
protected byte[] getRCHPacketForAuth(String qryString)
protected CRivRecord[] getRCHQueryRecords(String subject, Object qryData)
public void rchAddAmosListener(IRivRecordListener recListener)

```

```

public void rchAddModelListener(IRivRecordListener recListener)
public boolean rchClearEventRecord(CRivRecord eventRec)
public boolean rchDeleteModelRecord(CRivRecord modelRec)
public boolean rchDeleteProfileRecord(CRivRecord profileRec)
public boolean rchDeleteUserRecord(CRivRecord userRec)
public boolean rchInsertEventRecord(CRivRecord eventRec)
public boolean rchInsertModelRecord(CRivRecord modelRec)
public boolean rchInsertProfileRecord(CRivRecord profileRec)
public boolean rchInsertUserRecord(CRivRecord userRec)
public void rchRemoveAmosListener(IRivRecordListener recListener)
public void rchRemoveModelListener(IRivRecordListener recListener)
public boolean rchUpdateEventRecord(CRivRecord eventRec, CRivVarBindList evVbList)
public boolean rchUpdateModelRecord(CRivRecord modelRec, CRivVarBindList
modVbList)
public boolean rchUpdateProfileRecord(CRivRecord profileRec, CRivVarBindList
modVbList)
public boolean rchUpdateUserRecord(CRivRecord userRec, CRivVarBindList modVbList)

```

Class *CRivClientInterface*

Inner classes

```

CRivClientInterface.RCIRecordListener
CRivClientInterface.RCITimerCallback

```

Constructors

```

public CRivClientInterface()
public CRivClientInterface(CRivClient client)

```

Public instance methods

```

public CRivClient getRCIClient()
public int rciAddOnceOnlyTimer(long interval, IRivTimerCallback timerCallback)
public int rciAddRepeatTimer(long interval, IRivTimerCallback timerCallback)
public int rciAddService(String subject, IRivRecordListener userListener)
public boolean rciRemoveService(IRivRecordListener recListener)
public boolean rciRemoveTimer(IRivTimerCallback timerCallback)
public int rciSendToClient(String subject, Object data, IRivRecordListener
userListener)
public int rciTimedSendToClient(String subject, Object data, long timeout,
IRivRecordListener userListener)
public void setRCIClient(CRivClient client)

```

Class *CRivClientInterface.RCIRecordListener*

Constructor

```

public CRivClientInterface.RCIRecordListener (IRivRecordListener userListener,
boolean qryType)

```

Public instance methods

```

public void rrlRivRecordsReceived(CRivRecord[] rivRecs)
public void rrlTimeOutReceived()
public void setRCIRLDataHandler(CRivRvDataHandler rvHandler)

```

Class *CRivClientInterface.RCITimerCallback*

Constructor

```
public CRivClientInterface.RCITimerCallback(IRivTimerCallback timerCallback)
```

Public instance method

```
public void onTimer(COM.TIBCO.rv.RvTimer invoker)
```

Class *CRivDbEntity*

Constructors

```
public CRivDbEntity(CRivDbEntity dbEntity)
public CRivDbEntity(String dbName, String tblName, String colName)
```

Public instance methods

```
public String getRDEColName()
public String getRDEDbName()
public String getRDETblName()
public String toString()
```

Class *CRivDummyCallBack*

Constructor

```
public CRivDummyCallBack()
```

Public instance method

```
public void onData(String subject, COM.TIBCO.rv.RvSender replySender, Object data, COM.TIBCO.rv.RvListener invoker)
```

Class *CRivEvalClause*

Constructors

```
public CRivEvalClause(CRivEvalClause clauseToCopy)
public CRivEvalClause(int datatype, String text)
```

Public instance methods

```
public CRivAtom getRECCompileable()
public int getRECDataType()
public String toString()
```

Class *CRivException*

Inner class

```
static class CRivException.RivErrRec
```

Fields

```
public static final int CORE_NOT_PRESENT
public static final int CORRUPT_OBJECT_STORE
public static final int DUPLICATE_KEY_INSERT
public static final int DUPLICATE_OBJECT
public static final int DUPLICATE_PARENT
public static final int KEY_NOT_FOUND
public static final int NODE_HAS_ELEMENTAL
public static final int NODE_HAS_KIDS
public static final int NODE_MISSING
public static final int NULL_POINTER_REF
public static final int RIV_ERROR_BASE
public static final int RIV_EXIT
public static final int RIV_FATAL
public static final int RIV_INFO
public static final int RIV_OK
public static final int RIV_SICK
public static final int ROOT_EMPTY
public static final int RV_FAILED_TO_SEND
public static final int RV_INVALID_OUTBOUND_TYPE
public static final int RV_INVALID_SESSION
public static final int SUBTREE_EXCEPTION
public static final int TYPE_MISMATCH
public static final int UNDEFINED_ERROR
public static final int UNRESOLVED_FAULT_RULE
```

Constructors

```
public CRivException(int errorCode)
public CRivException(int errorCode, String text)
```

Public instance methods

```
public String getMessage()
public void rePrintErrorMessage(boolean printStackTrace)
public void rePrintErrorMessage(boolean printMessage, boolean printStackTrace)
```

Class *CRivException.RivErrRec*

Constructor

```
public CRivException.RivErrRec(int code, int level, String errorStr)
```

Public instance methods

```
public int getERCode()
public int getERLevel()
public String getERString()
```

Class *CRivExpr*

Fields

```
public static final int CLAUSE
public static final int IGNORED
public static final int INTERNAL
public static final int SUPPLIED
```

Constructors

```
public CRivExpr()
public CRivExpr(CRivAtom atmVal)
public CRivExpr(CRivDbEntity dbEntVal)
public CRivExpr(CRivEvalClause clauseVal)
public CRivExpr(CRivExpr exprVal)
```

Public instance methods

```
public CRivAtom getREAtomVal()
public CRivEvalClause getREClauseVal()
public CRivDbEntity getREEntityVal()
public int getREType()
public String toString()
```

Class *CRivFilter*

Constructors

```
public CRivFilter()
public CRivFilter(CRivFilter filterToCopy)
public CRivFilter(CRivFilter left, CRivFilter right, int oper)
public CRivFilter(CRivQueryAtom qryAtm)
```

Public instance methods

```
public CRivQueryAtom getRFKernel()
public CRivFilter getRFLeftSub()
public int getRFOperator()
public CRivFilter getRFRightSub()
public boolean rfAddLeftBranch(CRivFilter leftFilter)
public boolean rfAddRightBranch(CRivFilter rightFilter)
public boolean rfEvalFilter(CRivRecord record)
public static String rfPrint(CRivFilter filter)
public boolean rfReplaceBranch(CRivFilter oldBranch, CRivFilter newBranch)
public void setRFOperator(int oper)
public String toString()
```

Class *CRivFilterParser*

Constructor

```
public CRivFilterParser()
```

Public instance methods

```
public static int rfpAlgebraicToInt(String operString)
public static String rfpAlgebraicToString(int algebraic)
public static int rfpOperatorToInt(String operString)
public static String rfpOperatorToString(int oper)
public CRivFilter rfpParse(String text)
```

Class *CRivHashVector*

Constructors

```
public CRivHashVector()
public CRivHashVector(int initialCapacity)
```

Public instance methods

```
public void clear()
public Object clone()
public boolean containsKey(Object key)
public Object forcePut(Object key, Object value)
public Object get(Object key)
public int getRHVFirstIndexOfValue(Object value)
public int getRHVIndexOfKey(Object key)
public Object getRHVKeyAt(int index)
public Object getRHVValueAt(int index)
public Object put(Object key, Object value)
public void putAll(java.util.Map t)
public Object remove(Object key)
public Object rhvPutAt(Object key, Object value, int index)
```

Class *CRivMonitorFilterParser*

Constructor

```
public CRivMonitorFilterParser()
```

Public instance method

```
public CRivFilter rfpParse(String text)
```

Class *CRivQueryAtom*

Constructor

```
public CRivQueryAtom()
public CRivQueryAtom(CRivAtom name, int relOp, CRivAtom value)
public CRivQueryAtom(CRivExpr lhExpr, int relOp, CRivExpr rhExpr)
public CRivQueryAtom(CRivQueryAtom qryAtmToCopy)
```

Public instance methods

```
public CRivAtom getRQALeft()
public int getRQAOperator()
public CRivAtom getRQARight()
public int getRQAValueType()
public boolean rqaEvalQuery(CRivRecord record)
public void setRQALeft(CRivAtom name)
public void setRQAOperator(int oper)
public void setRQARight(CRivAtom value)
public String toString()
```

Class *CRivRecord*

Constructors

```
public CRivRecord()
public CRivRecord(CRivRecord recToCopy)
public CRivRecord(CRivVarBindList vbList)
```

Public instance methods

```
public boolean equals(Object obj)
public java.util.Enumeration getRRElements()
public int getRRSize()
public CRivAtom getRRValueOf(CRivAtom nameAtom)
public CRivAtom getRRValueOf(String name)
public CRivVarBind getRRValuePairOf(CRivAtom nameAtom)
public CRivVarBind getRRValuePairOf(String name)
public int rrAddValue(CRivAtom nm, CRivAtom val)
public int rrAddValue(CRivVarBind vb)
public int rrDecode(byte[] pktBytes)
public byte[] rrEncode()
public int rrInitFromVarList(CRivVarBindList vbList)
public String rrPrintVal(CRivAtom val, String pad)
public CRivVarBind rrRemoveValue(CRivAtom nameAtom)
public CRivVarBind rrRemoveValue(String name)
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value)
public CRivVarBind setRRValueOf(String name, CRivAtom value)
public String toString()
```

Class *CRivROMPer*

Fields

```
public static final int RIV_ROMP_ASN1
public static final int RIV_ROMP_BYTE_ARRAY
public static final int RIV_ROMP_FLOAT
public static final int RIV_ROMP_INT
public static final int RIV_ROMP_LIST_TYPE
public static final int RIV_ROMP_LONG
public static final int RIV_ROMP_LONGLONG
public static final int RIV_ROMP_OBJECT_TYPE
public static final int RIV_ROMP_STRING
```

Constructors

```
public CRivROMPer()
public CRivROMPer(byte[] bytes, boolean isBigEndian)
```

Instance methods

```
public CRivVarBind getRRNextAttrVal()
public byte[] getRRPacket()
public int getRRPacketLength()
public int rrAddAttrVal(CRivVarBind varbind)
public int rrAppendFrom(CRivROMPer romper)
protected CRivAtom rrDecodeASN1(byte[] pkt, int offset, int valueLength)
protected CRivAtom rrDecodeByteArray(byte[] pkt, int offset, int valueLength)
protected CRivAtom rrDecodeFloat(byte[] pkt, int offset, int valueLength)
protected CRivAtom rrDecodeInt(byte[] pkt, int offset, int valueLength)
protected CRivAtom rrDecodeLong(byte[] pkt, int offset, int valueLength)
protected String rrDecodeName(byte[] pkt, int offset)
protected CRivAtom rrDecodeString(byte[] pkt, int offset, int valueLength)
protected int rrDecodeType(byte[] pkt, int offset)
```



```

protected int rrDecodeValueLength(byte[] pkt, int offset)
protected void rrEncodeASN1(String asn1Str)
protected void rrEncodeByteArray(byte[] byteArr)
protected void rrEncodeFloat(float val)
protected void rrEncodeHeader(int type, String fieldName, int valLen)
protected void rrEncodeInt(int val)
protected void rrEncodeLong(long val)
protected void rrEncodeString(String str)
public void rrUpPacketSize(int increase)

```

Class *CRivRvDataHandler*

Fields

```

public static final int RRDH_DATA_RECEIVED
public static final int RRDH_TIMED_OUT
public static final int RRDH_UNKNOWN

```

Constructor

```

public CRivRvDataHandler(IRivRecordListener recListener)

```

Methods

```

protected java.lang.Thread getRRDHProcessorThread()
public int getRRDHStatus()
public void onData(String subject, COM.TIBCO.rv.RvSender replySender, Object data,
COM.TIBCO.rv.RvListener invoker)
public void onTimeOut(COM.TIBCO.rv.RvListener invoker)
public void rrdhCancelListener()
public CRivRecord[] rrdhConvertAtoms(CRivAtom[] byteArrayAtoms)
public void rrdhProcessData()
public void rrdhStartThread()
public void rrdhStopThread()
public void rrdhWait()
public void run()
public void setRRDHStatus(int status)

```

Class *CRivTransport*

Fields

```

public static final int RT_CLM_NOT_THERE
public static final int RT_DB_NOT_THERE
public static final int RT_INCOMPLETE_ROW
public static final int RT_INCORRECT_TYPE
public static final int RT_NO_ERROR
public static final int RT_PERM_DENIED
public static final int RT_SEARCH_FAILED
public static final int RT_TBL_NOT_THERE
public static final int SENT_PACKET_HEADER_LENGTH

```

Constructors

```

public CRivTransport()
public CRivTransport(boolean isBigEndian)

```

Public Instance/Class methods

```

public static byte[] getRTBytesForFloat(float floatVal, boolean bigEndian)
public static byte[] getRTBytesForInt(int intVal, boolean bigEndian)
public static byte[] getRTBytesForLong(long longVal, boolean bigEndian)
public java.util.Vector getRTDecodes()
public int getRTError()
public static float getRTFloatFromBytes(byte[] bytes, int offset, boolean bigEndian)
public static int getRTIntFromBytes(byte[] bytes, int offset, boolean bigEndian)
public static long getRTLLongFromBytes(byte[] bytes, int offset, boolean bigEndian)
public boolean isRTDone()
public void rtAddMessage(COM.TIBCO.rv.RvMsg msg)
public void rtDecodePktHeader(byte[] packet)
public void rtPrintError(String userText)
public void rtPurge()
public static float rtReadFloat(byte[] bytes)
public static int rtReadInt(byte[] bytes)
public static long rtReadLong(byte[] bytes)
public static byte[] rtSwapByteArray(byte[] byteArray)
public void setRTError(int error)

```

Class *CRivVarBind***Constructors**

```

public CRivVarBind(CRivVarBind arg)
public CRivVarBind(CRivAtom nm, CRivAtom vl)
public CRivVarBind(String nm, CRivAtom vl)
public CRivVarBind(String nm, String vl)
public CRivVarBind(String nm, int vl)
public CRivVarBind(String nm, long vl)

```

Public instance methods

```

public boolean equals(Object obj)
public CRivAtom getRVBName()
public String getRVBValAsString()
public CRivAtom getRVBValue()
public void setRVBName(CRivAtom name)
public void setRVBValue(CRivAtom value)
public String toString()

```

Class *CRivVarBindList***Fields**

```

protected int capacityIncrement
protected int elementCount
protected CRivVarBind[] elementData
protected boolean m_AllowDuplicates
protected transient int modCount
protected CRivAtom[] nameAtomData

```

Constructors

```

public CRivVarBindList()
public CRivVarBindList(int initialCapacity)
public CRivVarBindList(int initialCapacity, int capacityIncrement)

```

Public instance methods

```

public Object clone()
public boolean equals(Object Obj)
public int getRVBLCapacity()
public CRivVarBind getRVBLElementAt(int index)
public java.util.Enumeration getRVBLElements()
public CRivVarBind getRVBLFirstElement()
public int getRVBLIndexOf(CRivVarBind elem)
public int getRVBLIndexOf(CRivVarBind elem, int index)
public CRivVarBind getRVBLLastElement()
public int getRVBLLastIndexOf(CRivVarBind elem)
public int getRVBLLastIndexOf(CRivVarBind elem, int index)
public int getRVBLSize()
public CRivAtom getRVBLValue(CRivAtom nameAtom)
public CRivAtom getRVBLValue(String name)
public CRivVarBind getRVBLVarBind(CRivAtom nameAtom)
public CRivVarBind getRVBLVarBind(int index)
public CRivVarBind getRVBLVarBind(String name)
public boolean isRVBLEmpty()
public boolean rvblAdd(CRivVarBind vb)
public void rvblAdd(int index, CRivVarBind element)
public boolean rvblAddAll(CRivVarBindList vbList)
public boolean rvblAddAll(int index, CRivVarBindList vbList)
public void rvblAddElement(CRivVarBind vb)
public void rvblClear()
public boolean rvblContains(CRivVarBind elem)
public boolean rvblContainsAll(CRivVarBindList vbList)
public void rvblCopyInto(CRivVarBind[] anArray)
public void rvblEnsureCapacity(int minCapacity)
public void rvblInsertElementAt(CRivVarBind vb, int index)
public CRivVarBind rvblPut(CRivAtom name, CRivAtom value)
public boolean rvblRemove(CRivVarBind vb)
public CRivVarBind rvblRemove(int index)
public boolean rvblRemoveAll(CRivVarBindList vbList)
public void rvblRemoveAllElements()
public boolean rvblRemoveElement(CRivVarBind vb)
public void rvblRemoveElementAt(int index)
protected void rvblRemoveRange(int fromIndex, int toIndex)
public boolean rvblRetainAll(CRivVarBindList vbList)
public CRivVarBind[] rvblToArray()
public CRivVarBind[] rvblToArray(CRivVarBind[] a)
public void rvblTrimToSize()
public void setRVBLElementAt(CRivVarBind vb, int index)
public void setRVBLSize(int newSize)
public CRivVarBind setRVBLValueOf(CRivAtom name, CRivAtom value)
public CRivVarBind setRVBLVarBindAt(int index, CRivVarBind element)
public String toString()

```

Interface *IRivAlgebraic***Fields**

```

public static final int RA_AND
public static final int RA_NO_OP
public static final int RA_NOT
public static final int RA_OR

```

Interface *IRivConstants*

Fields

```

public static final int REC_ACK
public static final int REC_ALERT
public static final int REC_ASSIGN
public static final int REC_CLEAR
public static final int REC_CLEAR_ALL
public static final int REC_CLEAR_CHAIN
public static final int REC_DATA
public static final int REC_DEASSIGN
public static final int REC_DELETE
public static final int REC_EVENT
public static final int REC_NEW
public static final int REC_NONE
public static final int REC_TOOL
public static final int REC_UNACK
public static final int REC_UNDEFINED_ACTION
public static final int REC_UPDATE
public static final String RIV_AMOS_DN_CLAS
public static final String RIV_AMOS_DN_MOJO
public static final String RIV_AMOS_DN_TOPO
public static final String RIV_AMOS_DN_WHIP
public static final String RIV_AMOS_FN_ACKED
public static final String RIV_AMOS_FN_ACTN
public static final String RIV_AMOS_FN_ACTVE
public static final String RIV_AMOS_FN_ASSGN
public static final String RIV_AMOS_FN_CHTME
public static final String RIV_AMOS_FN_CONTACT
public static final String RIV_AMOS_FN_CONTO
public static final String RIV_AMOS_FN_CORRID
public static final String RIV_AMOS_FN_CRTME
public static final String RIV_AMOS_FN_EGRPID
public static final String RIV_AMOS_FN_EVID
public static final String RIV_AMOS_FN_EVTYPE
public static final String RIV_AMOS_FN_GLYPH
public static final String RIV_AMOS_FN_INTAC
public static final String RIV_AMOS_FN_NAME
public static final String RIV_AMOS_FN_OBJID
public static final String RIV_AMOS_FN_OCCRD
public static final String RIV_AMOS_FN_PART
public static final String RIV_AMOS_FN_SEVTY
public static final String RIV_AMOS_TN_ACLS
public static final String RIV_AMOS_TN_CONTS
public static final String RIV_AMOS_TN_ENTITY
public static final String RIV_AMOS_TN_EVENTS
public static final String RIV_AUTH_DB_EVENTS_OBJECT
public static final String RIV_AUTH_DB_PASSWORD
public static final String RIV_AUTH_DB_PERMISSIONS
public static final String RIV_AUTH_DB_PROFILE
public static final String RIV_AUTH_DB_PROFILES_TBL
public static final String RIV_AUTH_DB_RANK
public static final String RIV_AUTH_DB_USERNAME
public static final String RIV_AUTH_DB_USERS_OBJECT
public static final String RIV_AUTH_DB_USERS_TBL
public static final String RIV_AUTH_FIELD_CHANGE
public static final String RIV_AUTH_FIELD_CREATE
public static final String RIV_AUTH_FIELD_DELETE
public static final String RIV_AUTH_FIELD_EV_ACK
public static final String RIV_AUTH_FIELD_EV_ASSIGN
public static final String RIV_AUTH_FIELD_EV_CLEAR
public static final String RIV_AUTH_FIELD_EV_DEASSIGN

```

```

public static final String RIV_AUTH_FIELD_EV_UNACK
public static final String RIV_AUTH_FIELD_VIEW
public static final String RIV_AUTH_PASSWORD
public static final String RIV_AUTH_QUERY
public static final int RIV_AUTH_RANK_HIGHEST
public static final int RIV_AUTH_RANK_LOWEST
public static final String RIV_AUTH_USERNAME
public static final String RIV_MDL_DN_MSTR
public static final String RIV_MDL_DN_TRANS
public static final String RIV_MDL_FN_ACTN
public static final String RIV_MDL_FN_ACTVE
public static final String RIV_MDL_FN_ADDR
public static final String RIV_MDL_FN_CHTME
public static final String RIV_MDL_FN_CLASS
public static final String RIV_MDL_FN_CONTO
public static final String RIV_MDL_FN_CRTME
public static final String RIV_MDL_FN_DESCR
public static final String RIV_MDL_FN_EOID
public static final String RIV_MDL_FN_ETYPE
public static final String RIV_MDL_FN_NAME
public static final String RIV_MDL_FN_OBJID
public static final String RIV_MDL_FN_PART
public static final String RIV_MDL_FN_RELTO
public static final String RIV_MDL_FN_SECTY
public static final String RIV_MDL_FN_STATS
public static final String RIV_MDL_TN_CONTS
public static final String RIV_MDL_TN_EBNR
public static final String RIV_MDL_TN_ENTITY
public static final int SEV_CLEAR
public static final int SEV_CRITICAL
public static final int SEV_MAJOR
public static final int SEV_MINOR
public static final int SEV_NO_SEV
public static final int SEV_UNKNOWN
public static final int SEV_WARNING

```

Interface *IRivDataType*

Fields

```

public static final int RDT_ASN1
public static final int RDT_BYTE_ARRAY
public static final int RDT_FLOAT
public static final int RDT_INTEGER
public static final int RDT_LIST
public static final int RDT_LONG
public static final int RDT_OBJECT
public static final int RDT_STRING
public static final int RDT_UNDEFINED

```

Interface *IRivNodeType*

Fields

```

public static final int RNT_BRANCH
public static final int RNT_LEAF
public static final int RNT_ROOT

```

Interface *IRivOper*

Fields

```

public static final int RO_BETWEEN
public static final int RO_EQUALS
public static final int RO_EXISTS
public static final int RO_GREATER_THAN
public static final int RO_GREATER_THAN_OR_EQUAL
public static final int RO_IN
public static final int RO_IS_NOT_NULL
public static final int RO_IS_NULL
public static final int RO_LESS_THAN
public static final int RO_LESS_THAN_OR_EQUAL
public static final int RO_MATCHES
public static final int RO_NONE
public static final int RO_NOT_BETWEEN
public static final int RO_NOT_EQUALS
public static final int RO_NOT_IN
public static final int RO_NOT_MATCHES

```

Interface *IRivRecordListener*

Public instance methods

```

public void rrlRivRecordsReceived(CRivRecord[] rivRecs)
public void rrlTimeOutReceived()

```

Interface *IRivSubjects*

Fields

```

public static final String RIV_AUTH_DATA_STORE_SUBJ
public static final String RIV_AUTH_INTERNAL_SUBJ
public static final String RIV_AUTH_NOTIFY_SUBJ
public static final String RIV_AUTH_QUERY_SUBJ
public static final String RIV_CLASS_DATA_STORE_SUBJ
public static final String RIV_CLASS_INTERNAL_SUBJ
public static final String RIV_CLASS_NOTIFY_SUBJ
public static final String RIV_CLASS_QUERY_SUBJ
public static final String RIV_CTRL_NOTIFY_SUBJ
public static final String RIV_CTRL_QUERY_SUBJ
public static final String RIV_DISCO_NOTIFY_SUBJ
public static final String RIV_DISCO_QUERY_SUBJ
public static final String RIV_EVENT_DATA_STORE_SUBJ
public static final String RIV_EVENT_INTERNAL_SUBJ
public static final String RIV_EVENT_NOTIFY_SUBJ
public static final String RIV_EVENT_QUERY_SUBJ
public static final String RIV_MODEL_DATA_STORE_SUBJ
public static final String RIV_MODEL_INTERNAL_SUBJ
public static final String RIV_MODEL_NOTIFY_SUBJ
public static final String RIV_MODEL_QUERY_SUBJ
public static final String RIV_MONITOR_INTERNAL_SUBJ
public static final String RIV_MONITOR_NOTIFY_SUBJ
public static final String RIV_MONITOR_QUERY_SUBJ
public static final String RIV_MONITORAGENT_QUERY_SUBJ
public static final String RIV_MONITORAGENT_TOPODONE_SUBJ
public static final String RIV_REPOSIT_NOTIFY_SUBJ
public static final String RIV_REPOSIT_QUERY_SUBJ

```

Interface *IRivTimerCallback*

Method

```
public void rtcTimer()
```

Class *ParseException*

Fields

```
public Token currentToken
protected String eol
public int[][] expectedTokenSequences
protected boolean specialConstructor
public String[] tokenImage
```

Constructors

```
public ParseException()
public ParseException(String message)
public ParseException(Token currentTokenVal, int[][] expectedTokenSequencesVal, String[] tokenImageVal)
```

Methods

```
protected String add_escapes(String str)
```

Public instance methods

```
public String getMessage()
```

Class *RivFilter*

Fields

```
public Token jj_nt
public boolean lookingAhead
public Token token
public RivFilterTokenManager token_source
```

Constructors

```
public RivFilter(java.io.InputStream stream)
public RivFilter(java.io.Reader stream)
public RivFilter(RivFilterTokenManager tm)
```

Public instance methods

```
public final void disable_tracing()
public final void enable_tracing()
public final ParseException generateParseException()
public final void getArray()
public final CRivAtom getAtom()
public final java.util.Vector getAtomList()
public final CRivExpr getBetweenPredicate()
public final CRivDbEntity getColumnRef()
public final int getComparison()
public final CRivQueryAtom getComparisonPredicate(CRivExpr lhsExpr)
public final CRivFilter getCondition()
public final int getDataType()
public final CRivEvalClause getEvalRef()
public final CRivFilter getFilter()
```

```

public final CRivExpr getInPredicate()
public final CRivExpr getLikePredicate()
public final void getList()
public final CRivAtom getListWhole()
public final Token getNextToken()
public final void getObject()
public final CRivAtom getObjectWhole()
public final CRivQueryAtom getPredicate()
public final CRivExpr getScalarExp()
public final Token getToken(int index)
public final CRivVarBind getVarBind()
public final CRivVarBindList getVarBindList()
public final CRivAtom optEscape()
public final CRivFilter parseInput()
public CRivFilter parseInput(java.io.InputStream fileStream)
public CRivFilter parseInput(java.io.Reader reader)
public void ReInit(java.io.InputStream stream)
public void ReInit(java.io.Reader stream)
public void ReInit(RivFilterTokenManager tm)
public final CRivQueryAtom testForNull(CRivExpr lhsExpr)

```

Interface *RivFilterConstants*

Fields

```

public static final int DEFAULT
public static final int DIGIT
public static final int EOF
public static final int RIV_FLT_AND
public static final int RIV_FLT_APPROXNUM
public static final int RIV_FLT_ARRAY
public static final int RIV_FLT_BETWEEN
public static final int RIV_FLT_CHARACTER
public static final int RIV_FLT_DATA
public static final int RIV_FLT_EQ
public static final int RIV_FLT_ESCAPE
public static final int RIV_FLT_EVAL
public static final int RIV_FLT_FLOAT
public static final int RIV_FLT_GT
public static final int RIV_FLT_GTE
public static final int RIV_FLT_HAVING
public static final int RIV_FLT_IDENT
public static final int RIV_FLT_IN
public static final int RIV_FLT_INTEGER
public static final int RIV_FLT_INTNUM
public static final int RIV_FLT_IPADDRESS
public static final int RIV_FLT_IS
public static final int RIV_FLT_LIKE
public static final int RIV_FLT_LIST
public static final int RIV_FLT_LOCATOR
public static final int RIV_FLT_LONG
public static final int RIV_FLT_LONGLONG
public static final int RIV_FLT_LT
public static final int RIV_FLT_LTE
public static final int RIV_FLT_NEQ
public static final int RIV_FLT_NOT
public static final int RIV_FLT_NULLX
public static final int RIV_FLT_OBJECT
public static final int RIV_FLT_OR
public static final int RIV_FLT_SMALLINT
public static final int RIV_FLT_STRING
public static final int RIV_FLT_TEXT

```



```

public static final int RIV_FLT_TIME
public static final int RIV_FLT_TYPE
public static final int RIV_FLT_VARCHAR
public static final int RIV_FLT_WHERE
public static final String[] tokenImage

```

Class *RivFilterTokenManager*

Fields

```

protected char curChar
public static final String[] jjstrLiteralImages
public static final String[] lexStateNames

```

Constructors

```

public RivFilterTokenManager(ASCII_CharStream stream)
public RivFilterTokenManager(ASCII_CharStream stream, int lexState)

```

Public instance methods

```

public final Token getNextToken()
public void ReInit(ASCII_CharStream stream)
public void ReInit(ASCII_CharStream stream, int lexState)
public void SwitchTo(int lexState)

```

Class *RivMonitorFilter*

Fields

```

public Token jj_nt
public boolean lookingAhead
public Token token
public RivMonitorFilterTokenManager token_source

```

Constructors

```

public RivMonitorFilter(java.io.InputStream stream)
public RivMonitorFilter(java.io.Reader stream)
public RivMonitorFilter(RivMonitorFilterTokenManager tm)

```

Public instance methods

```

public final void disable_tracing()
public final void enable_tracing()
public final ParseException generateParseException()
public final void getArray()
public final CRivAtom getAtom()
public final java.util.Vector getAtomList()
public final CRivExpr getBetweenPredicate()
public final CRivDbEntity getColumnRef()
public final int getComparison()
public final CRivQueryAtom getComparisonPredicate(CRivExpr lhsExpr)
public final CRivFilter getCondition()
public final int getDataTypes()
public final CRivEvalClause getEvalRef()
public final CRivFilter getFilter()
public final CRivExpr getInPredicate()
public final CRivExpr getLikePredicate()
public final void getList()
public final CRivAtom getListWhole()
public final Token getNextToken()

```

```

public final void getObject()
public final CRivAtom getObjectWhole()
public final CRivQueryAtom getPredicate()
public final CRivExpr getScalarExp()
public final Token getToken(int index)
public final CRivVarBind getVarBind()
public final CRivVarBindList getVarBindList()
public final CRivAtom optEscape()
public final CRivFilter parseInput()
public CRivFilter parseInput(java.io.InputStream fileStream)
public CRivFilter parseInput(java.io.Reader reader)
public void ReInit(java.io.InputStream stream)
public void ReInit(java.io.Reader stream)
public void ReInit(RivMonitorFilterTokenManager tm)
public final CRivQueryAtom testForNull(CRivExpr lhsExpr)

```

Interface *RivMonitorFilterConstants*

Fields

```

public static final int DEFAULT
public static final int DIGIT
public static final int EOF
public static final int RIV_FLT_AND
public static final int RIV_FLT_APPROXNUM
public static final int RIV_FLT_ARRAY
public static final int RIV_FLT_BETWEEN
public static final int RIV_FLT_CHARACTER
public static final int RIV_FLT_DATA
public static final int RIV_FLT_EQ
public static final int RIV_FLT_ESCAPE
public static final int RIV_FLT_EVAL
public static final int RIV_FLT_FLOAT
public static final int RIV_FLT_GT
public static final int RIV_FLT_GTE
public static final int RIV_FLT_HAVING
public static final int RIV_FLT_IDENT
public static final int RIV_FLT_IN
public static final int RIV_FLT_INTEGER
public static final int RIV_FLT_INTNUM
public static final int RIV_FLT_IPADDRESS
public static final int RIV_FLT_IS
public static final int RIV_FLT_LIKE
public static final int RIV_FLT_LIST
public static final int RIV_FLT_LOCATOR
public static final int RIV_FLT_LONG
public static final int RIV_FLT_LONGLONG
public static final int RIV_FLT_LT
public static final int RIV_FLT_LTE
public static final int RIV_FLT_NEQ
public static final int RIV_FLT_NOT
public static final int RIV_FLT_NULLX
public static final int RIV_FLT_OBJECT
public static final int RIV_FLT_OR
public static final int RIV_FLT_SMALLINT
public static final int RIV_FLT_STRING
public static final int RIV_FLT_TEXT
public static final int RIV_FLT_TIME
public static final int RIV_FLT_TYPE
public static final int RIV_FLT_VARCHAR
public static final int RIV_FLT_WHERE
public static final String[] tokenImage

```

Class *RivMonitorFilterTokenManager*

Fields

```
protected char curChar
public static final String[] jjstrLiteralImages
public static final String[] lexStateNames
```

Constructors

```
public RivMonitorFilterTokenManager(ASCII_CharStream stream)
public RivMonitorFilterTokenManager(ASCII_CharStream stream, int lexState)
```

Public instance methods

```
public final Token getNextToken()
public void ReInit(ASCII_CharStream stream)
public void ReInit(ASCII_CharStream stream, int lexState)
public void SwitchTo(int lexState)
```

Class *Token*

Fields

```
public int beginColumn
public int beginLine
public int endColumn
public int endLine
public String image
public int kind
public Token next
public Token specialToken
```

Constructor

```
public Token()
```

Public instance methods

```
public static final Token newToken(int ofKind)
public final String toString()
```

Class *TokenMgrError*

Constructors

```
public TokenMgrError()
public TokenMgrError(boolean EOFSeen, int lexState, int errorLine, int
errorColumn, String errorAfter, char curChar, int reason)
public TokenMgrError(String message, int reason)
```

Method

```
protected static final String addEscapes (String str)
```

Summary

This chapter has detailed all of the classes and interfaces associated with the *Cisco Mobile Wireless Fault Mediator 2.0 - Java API Guide*.



Differences between Version 2 and NMOS Java API's

This chapter lists the classes and methods from the OpenRiver Version 2 Java API with their replacements in the MWFM NMOS Java API.

MWFM NMOS Java API replacements

Throughout this appendix the version 2 Java API class/method appears in **black** and the corresponding MWFM NMOS Java API class/method appears in *emphasis*.

Whereas the version 2 Java API used 3 separate packages, the MWFM NMOS Java API utilizes only one.

In most cases, the differences between the version 2 Java API and the MWFM NMOS Java API is a simple re-naming of the class/method. The main difference is that in the MWFM NMOS Java API there is no “Data Model” as no data is held by the API. Therefore, a user of the API would have to set up their own classes to store whatever data they were interested in. Whereas the version 2 Java API contained classes and methods which were specific to Events and Model data, the MWFM NMOS Java API contains only generic classes and methods. For example, the generic class *CRivRecord* replaces the specific **OrvEvRec** and **OrvModelInstance** version 2 classes. *CRivRecord* has also replaced **OrvPollDef**, **OrvUser**, and **OrvProfile**.

The version 2 Java API made use of specific methods such as **getAllEventRecords()**, whereas the MWFM NMOS Java API utilizes only generic methods which can be used to access any of the databases held by the MWFM NMOS Java processes. No specific “convenience” methods are supplied, so the user will need to know a number of specifics to carry out each task, such as the format of query strings, or the name of the database they want to insert a record into.

The orv_web.kernel package

V2 — orv_web.kernel.OrvASN1Address

V3 — com.riversoft.riv_web.vertigo.CRivASN1Address

```

public class OrvASN1Address extends Object {
    //Public Constructors
    public OrvASN1Address();
    public CRivASN1Address();

    public OrvASN1Address(int relativeAddress, int[] fullAddress, int addressDepth);
    public CRivASN1Address(int relativeAddress, int[] fullAddress, int addressDepth);

    public OrvASN1Address(OrvASN1Address parent, int child);
    public CRivASN1Address(CRivASN1Address parent, int child);

    public OrvASN1Address(OrvASN1Address base, OrvASN1Address addr);
    public CRivASN1Address raaUnion(CRivASN1Address base, CRivASN1Address qual);

    public OrvASN1Address(int firstNEntries, OrvASN1Address base);
    public CRivASN1Address(int firstNEntries, CRivASN1Address base);

    public OrvASN1Address(OrvASN1Address addressToCopy);
    public CRivASN1Address(CRivASN1Address addressToCopy);

    public OrvASN1Address(String stringRep);
    public CRivASN1Address(String stringRep);

    //Public Instance Methods
    public int getOrvAddressAt(int position);
    public int getRAAAddressAt(int position);

    public int getOrvAddressDepth();
    public int getRAADepth();

    public int getOrvHashASN1();
    Method no longer implemented in the RiverSoft NMOS Java. Simply use hashCode().

    public long getOrvHashedKey();
    Method no longer implemented in the RiverSoft NMOS Java API. Instead, use hashCode().

    public int getOrvRelativeAddress();
    public int getRAARelativeAddress();

    public int[] getOrvFullAddress();
    public int[] getRAAFullAddress();

    public boolean isOrvMatchAddress(OrvASN1Address compare);
    public boolean isRAAMatchAddress(CRivASN1Address compare);

    public boolean isOrvPartialMatch(OrvASN1Address compare);
    public boolean isRAAPartialMatch(CRivASN1Address compare);

    public String toString();
    public String toString();
}

```

V2—`orv_web.kernel.OrvAtom`

V3—`com.riversoft.riv_web.vertigo.CRivAtom`

```
public class OrvAtom extends Object {
```

//Constants

```
public static final int INTEGER, FLOAT, STRING;
public static final int DATE, VECTOR, HASHTABLE, OBJECT;
public static final int ORV_ASN1, ORV_UNDEFINED;
```

These "Atom type" constants are now defined in `IRivDataType`:

INTEGER - `IRivDataType.RDT_INTEGER`;

FLOAT - `IRivDataType.RDT_FLOAT`;

STRING - `IRivDataType.RDT_STRING`;

DATE - *No longer implemented. Dates likely to be RDT_LONG*;

VECTOR - `IRivDataType.RDT_LIST`;

HASHTABLE - *no longer implemented. Use - `IRivDataType.RDT_OBJECT`, a list of name/value pairs.*

OBJECT - `IRivDataType.RDT_OBJECT`;

ORV_ASN1 - `IRivDataType.RDT_ASN1`;

ORV_UNDEFINED - `IRivDataType.RDT_UNDEFINED`;

```
public static final int EQUALS, NOT_EQUALS;
public static final int LESS_THAN, LESS_THAN_OR_EQUAL;
public static final int GREATER_THAN;
public static final int GREATER_THAN_OR_EQUAL;
```

These Operator constants are now found in `IRivOper` as `RO_EQUALS`, `RO_NOT_EQUALS` etc.

//Public Constructors

```
public OrvAtom();
public CRivAtom();
```

```
public OrvAtom(int value);
public CRivAtom(int value);
```

```
public OrvAtom(float value);
public CRivAtom(float value);
```

```
public OrvAtom(String value);
public CRivAtom(String value);
```

```
public OrvAtom(Date value);
No longer implemented. Instead, use:
public CRivAtom(long value);
```

```
public OrvAtom(Vector value);
public CRivAtom(Vector value);
```

```
public OrvAtom(Hashtable value);
No longer implemented. Instead, use:
public CRivAtom(CRivVarBindList value);
```

```
public OrvAtom(OrvASN1Address value);
public OrvAtom(CRivASN1Address value);
```

```
public OrvAtom(OrvAtom atomToCopy);
public CRivAtom(CRivAtom atomToCopy);
```

//Public Instance Methods

```
public long dateValue();
No longer implemented. Replaced by:
public long getRALongValue();

public float floatValue();
public float getRAFloatValue();

public Class getOrvDataType();
public Class getRADataClass();

public int getOrvDataTypeAsInt();
public int getRAType();

public Hashtable hashtableValue();
No longer implemented. Replaced by:
public long getRAObject();

public int intValue();
public int getRAIntValue();

public boolean isAtomInVector(OrvAtom entry);
public boolean isRAAtomInVector(CRivAtom entry);

public OrvASN1Address orvASN1AddressValue();
public CRivASN1Address getRAASN1Address();

public boolean orvAtomEvalRelop(int operator, OrvAtom atomToCompare);
public boolean raEvalRelop(int operator, CRivAtom atomToCompare);

public void orvAtomSet(OrvASN1Address value);
public void setRAValue(CRivASN1Address value);

public void orvAtomSet(String value);
public void setRAValue(String value);

public void orvAtomSet(int value);
public void setRAValue(int value);

public void orvAtomSet(float value);
public void setRAValue(float value);

public void orvAtomSet(Date value);
No longer implemented. Replaced by:
public void setRAValue(long value);

public void orvAtomSet(Vector value);
public void setRAValue(Vector value);
```



```

public void orvAtomSet(Hashtable value);
No longer implemented. Replaced by:
public void setRAValue(Hashtable value);

public String toString();
public String toString();

public Vector vectorValue();
public Vector getRAVector();
}

```

V2—`orv_web.kernel.OrvCryptograph`

V3—Not implemented in the RiverSoft NMOS Java API.

```

public class OrvCryptograph extends Object {

//Public Instance Methods
public static String OCCryptNet(String str);
}

```

V2—`orv_web.kernel.OrvEvRec`

V3—`com.riversoft.riv_web.vertigo.CRivRecord`

The class `CRivRecord` is the fundamental stored item in ALL RiverSoft data engines. It replaces and obsoletes classes such as `OrvEvRec` and `OrvModelInstance` in the version 2 Java API.

```

public class OrvEvRec extends Object {

//Constants
public static final int CAUSE_TYPE_ROOT;
public static final int CAUSE_TYPE_SYMPTOM;
public static final int INTERNAL_ACTION_ACK;
public static final int INTERNAL_ACTION_ASSN;
public static final int INTERNAL_ACTION_BAD;
public static final int INTERNAL_ACTION_CLEAR;
public static final int INTERNAL_ACTION_CLEARALL;
public static final int INTERNAL_ACTION_DEACK;
public static final int INTERNAL_ACTION_DELETE;
public static final int INTERNAL_ACTION_NONE;
public static final int INTERNAL_ACTION_TOOL;
public static final int INTERNAL_ACTION_UPDATE ;
public static final int MGMT_NO_PROT, MGMT_PROT_ASCII;
public static final int MGMT_PROT_CMIP, MGMT_PROT_CMOT;
public static final int MGMT_PROT_ICMP, MGMT_PROT_OTHER;
public static final int MGMT_PROT_SNMP, MGMT_PROT_TRAP;
public static final String ORV_EVENT_FN_ALTID;
public static final String ORV_EVENT_FN_CAUSE;
public static final String ORV_EVENT_FN_CLSNM;
public static final String ORV_EVENT_FN_CONTACT;
public static final String ORV_EVENT_FN_CORRID;
public static final String ORV_EVENT_FN_COUNT;
public static final String ORV_EVENT_FN_CREATE;
public static final String ORV_EVENT_FN_EGRPID;
public static final String ORV_EVENT_FN_EVDESC;
public static final String ORV_EVENT_FN_EVID;

```

```

public static final String ORV_EVENT_FN_EVTYPE;
public static final String ORV_EVENT_FN_GLYPH;
public static final String ORV_EVENT_FN_INSTNAME;
public static final String ORV_EVENT_FN_INTAC;
public static final String ORV_EVENT_FN_LOCN;
public static final String ORV_EVENT_FN_MGMTPROT;
public static final String ORV_EVENT_FN_NAME;
public static final String ORV_EVENT_FN_REPEAT;
public static final String ORV_EVENT_FN_STATE;
public static final String ORV_EVENT_FN_SYSID;
public static final int SEV_CLEAR, SEV_CRITICAL;
public static final int SEV_MAJOR, SEV_MINOR, SEV_NO_SEV;
public static final int SEV_UNKNOWN, SEV_WARNING;
public static final int SYS_ID_AMOS,
public static final int TYPE_ALERT;
public static final int TYPE_DATA;
public static final int TYPE_EVENT;

```

None of the above constants are defined in the MWFM NMOS Java API.

//Public Constructors

```

public OrvEvRec(Vector values);
public CRivRecord(CRivVarBindList vbList);

public OrvEvRec(OrvEvRec objectToCopy);
public CRivRecord(CRivRecord vbList);

```

//Public Instance Methods

```

public synchronized void addEntry(String name, OrvAtom value);
public int rrAddValue(CRivAtom nm, CRivAtom val);
public int rrAddValue(CRivVarBind vb);

public synchronized String getInjectString (Hashtable overrides);
Not implemented in the RiverSoft NMOS Java API.

public synchronized int getOrvEventId();
Use public CRivAtom getRRValueOf("EventId");

public synchronized int getOrvEventType();
Use public CRivAtom getRRValueOf("EventType");

public synchronized int getOrvState();
Use public CRivAtom getRRValueOf("Severity");

public synchronized OrvAtom getOrvValueOf(String name);
public CRivAtom getRRValueOf(String name);

public synchronized Hashtable getOrvValues();
public Enumeration getRRElements();

public synchronized Vector getOrvValuesVector();
public Enumeration getRRElements();

public synchronized boolean setValueOf(String name, OrvAtom value);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);

public synchronized String toString();
public synchronized String toString();
}

```

V2—`orv_web.kernel.OrvModelInstance`

V3—`com.riversoft.riv_web.vertigo.CRivRecord`

The class `CRivRecord` is the fundamental stored item in ALL MWFM data engines. It replaces and obsoletes classes such as `OrvEvRec` and `OrvModelInstance` in the version 2 Java API.

```
public class OrvModelInstance extends Object {
```

//Constants

```
public static final String[] __mdl_tags;
public static final int[] __mdl_datatypes;
public static final int MDL_OBJNM, MDL_CLSNM, MDL_SYSDT;
public static final int MDL_LOC, MDL_PARENTS, MDL_NETADD;
    public static final int MDL_PHYSADD, MDL_NETPROT, MDL_TYPE;
    public static final int MDL_ACCESS, MDL_ACTION, MDL_LAYOUT;
    public static final int Tpo_Bus;
public static final int Tpo_None;
public static final int Tpo_NoSubPlane;
public static final int Tpo_Ring;
public static final int Tpo_SpecialStar;
public static final int Tpo_Star;
```

None of the above constants are defined in the MWFM NMOS Java API.

//Public Constructors

```
public OrvModelInstance();
public CRivRecord();

public OrvModelInstance(Vector values);
public CRivRecord(CRivVarBindList vbList);

public OrvModelInstance(OrvModelInstance objectToCopy);
public CRivRecord(CRivRecord recToCopy);

public OrvModelInstance(OrvAtom name, OrvAtom class, OrvAtom descr);
No longer implemented. Instead, use:
public CRivRecord(CRivVarBindList vbList);
```

//Public Instance Methods

```
public synchronized void addEntry(String name, OrvAtom value);
public int rrAddValue(CRivAtom nm, CRivAtom val);
public int rrAddValue(CRivVarBind vb);

public synchronized String getInjectString();
Not implemented in the RiverSoft NMOS Java API.

public synchronized OrvAtom getOrvDescription();
Use public CRivAtom getRRValueOf("Description");

public synchronized Vector getOrvIndices();
Not implemented in the RiverSoft NMOS Java API.

public synchronized OrvAtom getOrvInstClass();
Use public CRivAtom getRRValueOf("ClassName");
```

```

public synchronized OrvASN1Address getOrvLocator();
Not implemented in the RiverSoft NMOS Java API.

public synchronized OrvAtom getOrvName();
Use public CRivAtom getRRValueOf("EntityName");

public synchronized Vector getOrvParents();
Not implemented in the RiverSoft NMOS Java API.

public synchronized OrvAtom getOrvValueOf(String field);
public CRivAtom getRRValueOf(String name);

public synchronized OrvAtom getOrvValueOf(int fieldId);
Not implemented in the RiverSoft NMOS Java API.

public synchronized Hashtable getOrvValues();
public Enumeration getRRElements();

public synchronized Vector getOrvValuesVector();
public Enumeration getRRElements();

public synchronized boolean setValueOf(String name, OrvAtom value);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);

public synchronized String toString();
public synchronized String toString();
}

```

V2—`orv_web.kernel.OrvPollDef`

V3—`com.riversoft.riv_web.vertigo.CRivRecord`

No direct replacement for `OrvPollDef` in V3 API as yet. Use the class `CRivRecord`, the fundamental stored item in all Cisco data engines.

```

public class OrvPollDef extends Object {

//Constants
public static final int POLL_NM, POLL_PRO, POLL_COMM;
public static final int POLL_THRESH, POLL_TYPE, POLL_HOST;
public static final int POLL_STAT, POLL_DESC, POLL_EVNM;
public static final int POLL_LOC, POLL_CONTACT, POLL_IFINDEX;
public static final int POLL_IFAD, POLL_VBS, POLL_TAGS;
public static final int POLL_TRAPNM, POLL_FREQ, POLL_TIMEOUT;
public static final int POLL_EACHPOLL, POLL_RESTORE;
public static final int POLL_SUPPRULE, POLL_OBJNAME;
public static final int POLL_CLSNAME, POLL_ACTIVE;
public static final int POLL_ACTION, POLL_AGENTS;
public static final int POLL_CONTROL;
None of the above constants are as yet defined in the RiverSoft NMOS Java API.

//Public Constructors
public OrvPollDef(Vector values);
public CRivRecord(CRivVarBindList vbList);

public OrvPollDef(OrvPollDef objectToCopy);
public CRivRecord(CRivRecord recToCopy);
}

```

//Public Instance Methods

```

public synchronized void addEntry(String name, OrvAtom value);
public int rrAddValue(CRivAtom nm, CRivAtom val);
public int rrAddValue(CRivVarBind vb);

public synchronized OrvAtom getOrvPollName();
Use public CRivAtom getRRValueOf("PollName");

public synchronized OrvAtom getOrvValueOf(String name);
public CRivAtom getRRValueOf(String name);

public synchronized OrvAtom getOrvValueOf(int fieldId);
Not implemented in the RiverSoft NMOS Java API.

public synchronized Hashtable getOrvValues();
public Enumeration getRRElements();

public synchronized Vector getOrvValuesVector();
public Enumeration getRRElements();

public synchronized boolean setValueOf(String name, OrvAtom value);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);

public synchronized String toString();
public synchronized String toString();
}

```

V2—`orv_web.kernel.OrvVarBind`**V3—`com.riversoft.riv_web.vertigo.CRivVarBind`**

```

public class OrvVarBind extends Object {
    //Public Constructors

    public OrvVarBind(OrvAtom nm, OrvAtom vl);
    public CRivVarBind(CRivAtom nm, CRivAtom vl);

    public OrvVarBind(OrvVarBind arg);
    public CRivVarBind(CRivVarBind arg);

    //Public Instance Methods

    public OrvAtom getOrvOverride();
    Not implemented in the RiverSoft NMOS Java API.

    public OrvAtom getOrvName();
    public CRivAtom getRVBName();

    public OrvAtom getOrvValue();
    public CRivAtom getRVBValue();

    public String getOrvValAsString();
    public String getRVBValAsString();

    public boolean setOrvOverride(OrvAtom value);
}

```

Not implemented in the RiverSoft NMOS Java API.

```
public boolean setOrvValue(OrvAtom arg);
public boolean setRVBValue(CRivAtom val);

}
```

The orv_web.network package

V2—`orv_web.network.OrvAuthMeta`

V3—Not being implemented in RiverSoft NMOS Java API. `CRivRecord` is used in place of `OrvUser` and `OrvProfile`.

```
public class OrvAuthMeta extends Object {
    // Constants
    public static final int AuthUndef, AuthUser, AuthProfile;
    //Public Constructors
    public OrvAuthMeta();
    public OrvAuthMeta(int type);
    //Public Instance Methods
    public int getOrvType();
    public void setOrvType(int type);
}
```

V2—`orv_web.network.OrvClient`

V3—`com.riversoft.riv_web.vertigo.CRivClient`

```
public class OrvClient extends Object {

    // Constants
    public static final String ORV_AUTH_QUERY_SUBJ;
    public static final String ORV_AUTH_INTERNAL_SUBJ;
    public static final String ORV_AUTH_NOTIFY_SUBJ;
    public static final String ORV_AUTH_DATASTORE_SUBJ;
    public static final String ORV_EVENT_QUERY_SUBJ;
    public static final String ORV_EVENT_INTERNAL_SUBJ;
    public static final String ORV_EVENT_NOTIFY_SUBJ;
    public static final String ORV_EVENT_DATA_STORE_SUBJ;
    public static final String ORV_MODEL_QUERY_SUBJ;
    public static final String ORV_MODEL_INTERNAL_SUBJ;
    public static final String ORV_MODEL_NOTIFY_SUBJ;
    public static final String ORV_MODEL_DATA_STORE_SUBJ;
    public static final String ORV_REQT_QUERY_SUBJ;
    public static final String ORV_REQT_INTERNAL_SUBJ;
    public static final String ORV_REQT_NOTIFY_SUBJ;
    public static final String ORV_REQT_DATASTORE_SUBJ;
    None of the above constants are defined in the RiverSoft NMOS Java API.
}
```

//Public Constructors

```
public OrvClient(long timeOut, String domain);
public CRivClient(long timeOut, String domain);

public OrvClient(String domain);
Use public CRivClient(long timeOut, String domain);
```

//Public Instance Methods

```

public String getOrvDomain();
public String getRCDomain();

public String getOrvDomainSubj(String subjStub);
public String getRCDomainSubj(String subjStub);

public long getOrvLatency();
public long getRCTimeOut ();

public RvSession getOrvSession();
Not implemented in the RiverSoft NMOS Java API.

public boolean openSession();
public int rcInitSession();

public boolean openSession(String hostname, int port);
public int rcInitSession(String hostname, int port);

public boolean openSession(String hostname);
public int rcInitSession(String hostname);

public void orvAddListener(RvDataCallback handler, String subject);
public int rcAddService(String subject, RvDataCallback handler);

public void orvAddTimer(int when, RvTimerCallback handler);
public RvTimer rcAddOnceOnlyTimer(long interval, RvTimerCallback handler);

public void orvAddRepeatingTimer(long frequency, RvTimerCallback handler);
public RvTimer rcAddRepeatTimer(long interval, RvTimerCallback handler);

public void orvSend(String subject, Object data);
public int rcSend(String subject, Object data);

public void orvSend(String subject, Object data, RvDataCallback replyHandler);
public int rcSend(String subject, Object data, RvDataCallback replyHandler);

public void orvTimedSend(String subject, Object data, RvDataCallback
replyHandler, long timeOut);
public int rcTimedSend(String subject, Object data, long timeOut, RvDataCallback
replyHandler);

public void terminateOrvClient();
public int rcTerminateSession();

}

```

V2—`orv_web.network.OrvClientInterface`**V3—`com.riversoft.riv_web.vertigo.CRivClientInterface`**

```

public class OrvClientInterface extends Object {

//Public Constructors

public OrvClientInterface(OrvClient clientObject);
public CRivClientInterface(CRivClient clientObject);

```

//Public Instance Methods

```

public void addAmosEventListener (AmosEventListener l);
Use the following method with the appropriate subject:
public void rciAddService (String subject, IRivRecordListener recListener);

public void addReqtEventListener (ReqtEventListener l);
Use the following method with the appropriate subject:-
public void rciAddService (String subject, IRivRecordListener recListener);

public void addTopoEventListener (TopoEventListener l);
Use the following method with appropriate subject:
public void rciAddService (String subject, IRivRecordListener recListener);

public Vector constructRecordsFromRvList (Vector RvMsgList);
Not implemented in the RiverSoft NMOS Java API.

public Vector constructVarBindsListFromRvMsg (RvMsg rvMessage);
Not implemented in the RiverSoft NMOS Java API.

public boolean createEventRecord (OrvEvRec evRec);
Use the following method with appropriate subject:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public boolean createModelInstance (OrvModelInstance modelInstance);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public boolean deleteModelInstance (OrvModelInstance modelInstance);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getAllEventRecords ();
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getAllModelInstances ();
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getAllPolls ();
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getAuthProfiles (String userName, String passWord);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getAuthUsers (String userName, String passWord);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getEventRecords (OrvEvRec queryEvRec);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient (String subject, String data, long timeout,
IRivRecordListener userListener);

```



```

public Vector getInterfaces(OrvModelInstance modelInstance);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient(String subject, String data, long timeout,
IRivRecordListener userListener);

public Vector getModelInstances(OrvModelInstance queryModelInstance);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient(String subject, String data, long timeout,
IRivRecordListener userListener);

public OrvClient getOrvClient();
public CRivClient getRCIClient();

public Vector getPolls(OrvPollDef queryPollDef);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient(String subject, String data, long timeout,
IRivRecordListener userListener);

public boolean getProfileDataReceived();
Not implemented in the RiverSoft NMOS Java API.

public Vector getUpdateEventRecords();
Not implemented in the RiverSoft NMOS Java API.

public Vector getUpdateModelRecords();
Not implemented in the RiverSoft NMOS Java API.

public boolean getUserDataReceived();
Not implemented in the RiverSoft NMOS Java API.

public boolean modifyAlertRecord(OrvEvRec old, OrvEvRec new);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient(String subject, String data, long timeout,
IRivRecordListener userListener);

public boolean modifyModelInstance(OrvModelInstance old, OrvModelInstance new);
Use the following method with appropriate subject and data:
public int rciTimedSendToClient(String subject, String data, long timeout,
IRivRecordListener userListener);

public void notifyEventUpdates(RvDataCallback handler);
Not implemented in the RiverSoft NMOS Java API.

public void notifyModelUpdates(RvDataCallback handler);
Not implemented in the RiverSoft NMOS Java API.

public void removeAmosEventListener (AmosEventListener l);
public boolean rciRemoveService (IRivRecordListener recListener);

public void removeReqEventListener (ReqEventListener l);
public boolean rciRemoveService (IRivRecordListener recListener);
public void removeTopoEventListener (TopoEventListener l);
public boolean rciRemoveService (IRivRecordListener recListener);

public boolean restartPoll(OrvPollDef pollDef);
Not implemented in the RiverSoft NMOS Java API.

public void setEventDataReceived(boolean finished);
Not implemented in the RiverSoft NMOS Java API.

public void setEventRecords(Vector RvMsgList);
Not implemented in the RiverSoft NMOS Java API.

```

```

public void setModelDataReceived(boolean finished);
Not implemented in the RiverSoft NMOS Java API.

public void setProfileDataReceived(boolean value);
Not implemented in the RiverSoft NMOS Java API.

public void setProfiles(Vector profiles);
Not implemented in the RiverSoft NMOS Java API.

public void setUserDataReceived(boolean value);
Not implemented in the RiverSoft NMOS Java API.

public void setUsers(Vector users);
Not implemented in the RiverSoft NMOS Java API.

public boolean suspendPoll(OrvPollDef pollDef);
Not implemented in the RiverSoft NMOS Java API.

public void updateEventRecords(Vector RvMsgList);
Not implemented in the RiverSoft NMOS Java API.

public void updateModelRecords(Vector RvMsgList);
Not implemented in the RiverSoft NMOS Java API.

}

```

V2—`orv_web.network.OrvDataCallback`

V3—`com.riversoft.riv_web.vertigo.CRivRvDataHandler`

```

public class OrvDataCallback extends Object implements RvDataCallback,
RvTimeoutCallback, Runnable {

```

//Public Constructors

```

public OrvDataCallback(OrvClientInterface clientInt);
public CRivRvDataHandler(IRivRecordListener recListener);

public OrvDataCallback(OrvClientInterface clientInt, boolean updateType);
public CRivRvDataHandler(IRivRecordListener recListener);

```

//Public Instance Methods

```

public synchronized void addAmosEventListener (AmosEventListener l);
Not implemented in the RiverSoft NMOS Java API - each CRivRvDataHandler is passed its single IRivRecordListener in the constructor.

public synchronized void addReqEventEventListener (ReqEventEventListener l);
Not implemented in the RiverSoft NMOS Java API - each CRivRvDataHandler is passed its single IRivRecordListener in the constructor.

public synchronized void addTopoEventListener (TopoEventListener l);
Not implemented in the RiverSoft NMOS Java API - each CRivRvDataHandler is passed its single IRivRecordListener in the constructor.

```

```

public Vector getQueryData();
Not implemented in the RiverSoft NMOS Java API.

public boolean getTimedOut();
public int getRRDHStatus();

public boolean isOrvTransportDone();
public int getRRDHStatus();

public boolean isUpdateType();
Not implemented in the RiverSoft NMOS Java API.

public void onData(String subject, RvSender replySender, Object data, RvListener
listener);
public void onData(String subject, RvSender replySender, Object data, RvListener
listener);

public void onTimeOut(RvListener invoker);
public void onTimeOut(RvListener invoker);

public synchronized void removeAmosEventListener (AmosEventListener l);
Not implemented in the RiverSoft NMOS Java API.

public synchronized void removeReqEventListener (ReqEventListener l);
Not implemented in the RiverSoft NMOS Java API.

public void removeSubjectListener();
public void rrdhCancelListener();

public synchronized void removeTopoEventListener (TopoEventListener l);
Not implemented in the RiverSoft NMOS Java API.

public void run();
public void run();

public void startDataThread();
public void rrdhStartThread();

public void stopDataThread();
public void rrdhStopThread();
}

```

V2—`orv_web.network.OrvProfile`

V3—`com.riversoft.riv_web.vertigo.CRivRecord`

No direct replacement for `OrvProfile` anticipated for the RiverSoft NMOS Java API. Use the class `CRivRecord`, the fundamental stored item in all RiverSoft data engines.

```

public class OrvProfile extends OrvAuthMeta {

//Constants

public static final short ORV_AUTH_VIEW_BLANK;
public static final short ORV_AUTH_VIEW_FULL;
public static final short ORV_AUTH_VIEW_VIEW;
public static final short ORV_AUTH_VIEW_CHANGE;
public static final short ORV_AUTH_VIEW_CREATE;
public static final short ORV_AUTH_VIEW_DESTROY;

```

```

public static final short ORV_AUTH_EV_ACT_BLANK;
public static final short ORV_AUTH_EV_ACT_FULL;
public static final short ORV_AUTH_EV_ACT_ASSIGN;
public static final short ORV_AUTH_EV_ACT_ACK;
public static final short ORV_AUTH_EV_ACT_DEACK;
public static final short ORV_AUTH_EV_ACT_CLEAR;
public static final int ORV_AUTH_LOWEST_RANK;
public static final int ORV_AUTH_HIGHEST_RANK;
public static final int ACTION_EV_ASSIGN, ACTION_EV_ACK;
public static final int ACTION_EV_DEACK, ACTION_EV_CLEAR;
public static final int ACTION_VIEW_VIEW, ;
public static final int ACTION_VIEW_CHANGE;
public static final int ACTION_VIEW_CREATE;
public static final int ACTION_VIEW_DESTROY;
public static final int AUTH_NUM_CATEGORIES;
public static final int AUTH_EVENT_LIST, AUTH_ALERT_LIST;
public static final int AUTH_MAP_VIEW, AUTH_GRAPH_VIEW;
public static final int AUTH_HRTBEAT VIEW, AUTH_USER_ADMIN;
public static final int AUTH_COMMENTS, AUTH_POLL_DEFN;

```

None of the above constants are defined in the the RiverSoft NMOS Java API.

//Public Constructors

```

public OrvProfile(String name);
public CRivRecord(CRivVarBindList vbList);

public OrvProfile(String name, int type);
public CRivRecord(CRivVarBindList vbList);

public OrvProfile(OrvProfile orvProf);
public CRivRecord(CRivRecord recToCopy);

```

//Public Instance Methods

```

public OrvVarBind getOrvAuthority();
Not implemented in the RiverSoft NMOS Java API.

public short getOrvEvActMask();
Not implemented in the RiverSoft NMOS Java API.

public short getOrvMask(int category);
Not implemented in the RiverSoft NMOS Java API.

public String getOrvName();
public CRivAtom getRRValueOf(String name);
where name is "Profile".

public int getOrvRank();
public CRivAtom getRRValueOf(String name);
where name is "Rank".

public void orvAddPerm(short mask);
Not implemented in the RiverSoft NMOS Java API.

public void orvAddPerm(int category, short mask);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
where name is "Permissions".

public void orvAddPermToAll(short mask);
Not implemented in the RiverSoft NMOS Java API.

```

```

public boolean orvAllows(int category, int action);
Not implemented in the RiverSoft NMOS Java API.

public boolean orvAllows(int action);
Not implemented in the RiverSoft NMOS Java API.

public OrvProfile orvCreateActionProfile(int type);
Not implemented in the RiverSoft NMOS Java API.

public void orvDenyPerm(short mask);
Not implemented in the RiverSoft NMOS Java API.

public void orvDenyPerm(int category, short mask);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
where name is "Permissions".

public boolean orvPflsEqual(OrvProfile profToCompare);
public boolean equals(Object obj);

public void orvProfileCreate();
Not implemented in the RiverSoft NMOS Java API.

public boolean orvTestAllowed(short testMask, int action);
Not implemented in the RiverSoft NMOS Java API.

public void parseEvRecPerms(String permissions);
Not implemented in the RiverSoft NMOS Java API.

public void parseNetPerms(int action, String permissions);
Not implemented in the RiverSoft NMOS Java API.

public void setOrvAuth(OrvVarBind authorityVb);
Not implemented in the RiverSoft NMOS Java API.

public void setOrvRank(int rank);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
where name is "Rank".
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);
where name new CRivAtom("Rank").

}

```

V2—`orv_web.network.OrvTransport`

V3—`com.riversoft.riv_web.vertigo.CRivTransport`

```

public class OrvTransport extends Object {

//Public Constructors

public OrvTransport (OrvClient clientObj);
public CRivTransport ();

```

//Public Instance Methods

```

public Vector getOrvTransDecoded();
public Vector getRTDecodes();

public boolean isOrvTransDone();
public boolean isRTDone();

public void orvTransAddMessage(RvMsg data);
public void rtAddMessage(RvMsg data);

public void orvTransPurge();
public void rtPurge();
}

```

V2—`orv_web.network.OrvUser`**V3—`com.riversoft.riv_web.vertigo.CRivRecord`**

No direct replacement for OrvUser anticipated for the RiverSoft NMOS Java API. Use the class CRivRecord, the fundamental stored item in all RiverSoft data engines.

```
public class OrvUser extends OrvAuthMeta {
```

//Public Constructors

```

public OrvUser ();
public CRivRecord();

public OrvUser(int type);
public CRivRecord();

public OrvUser(OrvUser userToCopy);
public CRivRecord(CRivRecord recToCopy);

```

//Public Instance Methods

```

public OrvVarBind getOrvAuthority();
Not implemented in the RiverSoft NMOS Java API.

public String getOrvPassword();
public CRivAtom getRRValueOf(String name);
where name "Password".

public String getOrvProfile();
public CRivAtom getRRValueOf(String name);
where name is "Profile".

public String getOrvUsername();
public CRivAtom getRRValueOf(String name);
where name "Name".

public boolean orvUsrsEqual(OrvUser userToCompare);
public boolean equals(Object obj);

public boolean setOrvAuth(OrvVarBind authVb);
Not implemented in the RiverSoft NMOS Java API.

```

```

public void setOrvPassword(String passwd);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
where name is "Password".
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);
where name is new CRivAtom("Password").

public void setOrvProfile(String prof);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
where name is "Profile".
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);
where name is new CRivAtom("Profile").

public void setOrvUsername(String usernm);
public CRivVarBind setRRValueOf(String name, CRivAtom value);
where name is "Name".
public CRivVarBind setRRValueOf(CRivAtom name, CRivAtom value);
where name is new CRivAtom("Name").
}

```

The orv_web.network.event package

V2—`orv_web.network.event.AmosEventListener`

V3—`com.riversoft.riv_web.vertigo.IRivRecordListener`

```

public abstract interface AmosEventListener extends OrvEventListener
{

```

//Public Instance Methods

```

public abstract void eventRecordReceived(OrvEvRec evRec);
public abstract void rrlRivRecordsReceived(CRivRecord[] rivRecs);
}

```

V2—`orv_web.network.event.OrvEventListener`

V3—No base interface for `IRivRecordListener`.

```

public interface OrvEventListener{
}

```

V2—`orv_web.network.event.ReqtEventListener`

V3—`com.riversoft.riv_web.vertigo.IRivRecordListener`

```

public abstract interface ReqtEventListener extends OrvEventListener
{

```

//Public Instance Methods

```

public abstract void pollRecordReceived(OrvPollDef pollDef);
public abstract void rrlRivRecordsReceived(CRivRecord[] rivRecs);
}

```

V2—`orv_web.network.event.TopoEventListener`

V3—`com.riversoft.riv_web.vertigo.IRivRecordListener`

```
public abstract interface TopoEventListener extends OrvEventListener
{
```

//Public Instance Methods

```
public abstract void modelRecordReceived (OrvModelInstance modelInst)
public abstract void rrlRivRecordsReceived(CRivRecord[] rivRecs);
}
```