| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV PACKAGE**](http://docs.google.com/javax/sql/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/sql/rowset/serial/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sql/rowset/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package javax.sql.rowset

Standard interfaces and base classes for JDBC RowSet implementations.

**See:**

[**Description**](#3znysh7)

| **Interface Summary** | |
| --- | --- |
| [**CachedRowSet**](http://docs.google.com/javax/sql/rowset/CachedRowSet.html) | The interface that all standard implementations of CachedRowSet must implement. |
| [**FilteredRowSet**](http://docs.google.com/javax/sql/rowset/FilteredRowSet.html) | The standard interface that all standard implementations of FilteredRowSet must implement. |
| [**JdbcRowSet**](http://docs.google.com/javax/sql/rowset/JdbcRowSet.html) | The standard interface that all standard implementations of JdbcRowSet must implement. |
| [**Joinable**](http://docs.google.com/javax/sql/rowset/Joinable.html) | 1.0 Background |
| [**JoinRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html) | The JoinRowSet interface provides a mechanism for combining related data from different RowSet objects into one JoinRowSet object, which represents an SQL JOIN. |
| [**Predicate**](http://docs.google.com/javax/sql/rowset/Predicate.html) | The standard interface that provides the framework for all FilteredRowSet objects to describe their filters. |
| [**WebRowSet**](http://docs.google.com/javax/sql/rowset/WebRowSet.html) | The standard interface that all implementations of a WebRowSet must implement. |

| **Class Summary** | |
| --- | --- |
| [**BaseRowSet**](http://docs.google.com/javax/sql/rowset/BaseRowSet.html) | An abstract class providing a RowSet object with its basic functionality. |
| [**RowSetMetaDataImpl**](http://docs.google.com/javax/sql/rowset/RowSetMetaDataImpl.html) | Provides implementations for the methods that set and get metadata information about a RowSet object's columns. |

| **Exception Summary** | |
| --- | --- |
| [**RowSetWarning**](http://docs.google.com/javax/sql/rowset/RowSetWarning.html) | An extension of SQLException that provides information about database warnings set on RowSet objects. |

## Package javax.sql.rowset Description

Standard interfaces and base classes for JDBC RowSet implementations. This package contains interfaces and classes that a standard RowSet implementation either implements or extends.

## Table of Contents

* [1.0 Package Specification](#_2et92p0)
* [2.0 Standard RowSet Definitions](#_tyjcwt)
* [3.0 Implementater's Guide](#3dy6vkm)
* [4.0 Related Specifications](#1t3h5sf)
* [5.0 Related Documentation](#4d34og8)

### 1.0 Package Specification

This package specifies five standard JDBC RowSet interfaces. All five extend the [RowSet](http://docs.google.com/RowSet.html) interface described in the JDBC 3.0 specification. It is anticipated that additional definitions of more specialized JDBC RowSet types will emerge as this technology matures. Future definitions *should* be specified as subinterfaces using inheritance similar to the way it is used in this specification.

*Note:* The interface definitions provided in this package form the basis for all compliant JDBC RowSet implementations. Vendors and more advanced developers who intend to provide their own compliant RowSet implementations should pay particular attention to the assertions detailed in specification interfaces.

### 2.0 Standard RowSet Definitions

* [**JdbcRowSet**](http://docs.google.com/JdbcRowSet.html) - A wrapper around a ResultSet object that makes it possible to use the result set as a JavaBeansTM component. Thus, a JdbcRowSet object can be a Bean that any tool makes available for assembling an application as part of a component based architecture . A JdbcRowSet object is a connected RowSet object, that is, it **must** continually maintain its connection to its data source using a JDBC technology-enabled driver ("JDBC driver"). In addition, a JdbcRowSet object provides a fully updatable and scrollable tabular data structure as defined in the JDBC 3.0 specification.
* [**CachedRowSetTM>**](http://docs.google.com/CachedRowSet.html) **- A CachedRowSet object is a JavaBeansTM component that is scrollable, updatable, serializable, and generally disconnected from the source of its data. A CachedRowSet object typically contains rows from a result set, but it can also contain rows from any file with a tabular format, such as a spreadsheet. CachedRowSet implementations must use the SyncFactory to manage and obtain pluggable SyncProvider objects to provide synchronization between the disconnected RowSet object and the originating data source. Typically a SyncProvider implementation relies upon a JDBC driver to obtain connectivity to a particular data source. Further details on this mechanism are discussed in the** [**javax.sql.rowset.spi**](http://docs.google.com/spi/package-summary.html) **package specification.**
* [**WebRowSet**](http://docs.google.com/WebRowSet.html) **- A WebRowSet object is an extension of CachedRowSet that can read and write a RowSet object in a well formed XML format. This class calls an** [**XmlReader**](http://docs.google.com/spi/XmlReader.html) **object (an extension of the** [**RowSetReader**](http://docs.google.com/RowSetReader.html) **interface) to read a rowset in XML format. It calls an** [**XmlWriter**](http://docs.google.com/spi/XmlWriter.html) **object (an extension of the** [**RowSetWriter**](http://docs.google.com/RowSetWriter.html) **interface) to write a rowset in XML format. The reader and writer required by WebRowSet objects are provided by the SyncFactory in the form of SyncProvider implementations. In order to ensure well formed XML usage, a standard generic XML Schema is defined and published at**  [**http://java.sun.com/xml/ns/jdbc/webrowset.xsd**](http://java.sun.com/xml/ns/jdbc/webrowset.xsd)**.**
* [**FilteredRowSet**](http://docs.google.com/FilteredRowSet.html) **- A FilteredRowSet object provides filtering functionality in a programmatic and extensible way. There are many instances when a RowSet object has a need to provide filtering in its contents without sacrificing the disconnected environment, thus saving the expense of having to create a connection to the data source. Solutions to this need vary from providing heavyweight full scale SQL query abilities, to portable components, to more lightweight approaches. A FilteredRowSet object consumes an implementation of the** [**Predicate**](http://docs.google.com/Predicate.html) **interface, which may define a filter at run time. In turn, a FilteredRowSet object is tasked with enforcing the set filter for both inbound and outbound read and write operations. That is, all filters can be considered as bi-directional. No standard filters are defined; however, sufficient mechanics are specified to permit any required filter to be implemented.**
* [**JoinRowSet**](http://docs.google.com/JoinRowSet.html) **- The JoinRowSet interface describes a mechanism by which relationships can be established between two or more standard RowSet implementations. Any number of RowSet objects can be added to a JoinRowSet object provided the RowSetobjects can be related in a SQL JOIN like fashion. By definition, the SQL JOIN statement is used to combine the data contained in two (*or more*) relational database tables based upon a common attribute. By establishing and then enforcing column matches, a JoinRowSet object establishes relationships between RowSet instances without the need to touch the originating data source.**

### **3.0 Implementer's Guide**

**Compliant implementations of JDBC RowSet Implementations must follow the assertions described in this specification. In accordance with the terms of the** [**Java Community Process**](http://www.jcp.org)**, a Test Compatibility Kit (TCK) can be licensed to ensure compatibility with the specification. The following paragraphs outline a number of starting points for implementers of the standard JDBC RowSet definitions. Implementers should also consult the *Implementer's Guide* in the** [**javax.sql.rowset.spi**](http://docs.google.com/spi/package-summary.html) **package for guidelines on** [**SyncProvider**](http://docs.google.com/spi/SyncProvider.html) **implementations.**

**3.1 Role of the BaseRowSet Class**  
**A compliant JDBC RowSet implementation must implement one or more standard interfaces specified in this package and and may extend the** [**BaseRowSet**](http://docs.google.com/BaseRowSet.html) **abstract class. For example, a CachedRowSet implementation must implement the CachedRowSet interface and extend the BaseRowSet abstract class. The BaseRowSet class provides the standard architecture on which all RowSet implementations should be built, regardless of whether the RowSet objects exist in a connected or disconnected environment. The BaseRowSet abstract class provides any RowSet implementation with its base functionality, including property manipulation and event notification that is fully compliant with** [**JavaBeans**](http://java.sun.com/products/javabeans) **component requirements. As an example, all implementations provided in the reference implementations (contained in the com.sun.rowset package) use the BaseRowSet class as a basis for their implementations.**  
**The following table illustrates the features that the BaseRowSet abstract class provides.**

| **Feature** | **Details** |
| --- | --- |
| **Properties** | **Provides standard JavaBeans property manipulation mechanisms to allow applications to get and set RowSet command and property values. Refer to the documentation of the javax.sql.RowSet interface (available in the JDBC 3.0 specification) for more details on the standard RowSet properties.** |
| **Event notification** | **Provides standard JavaBeans event notifications to registered event listeners. Refer to the documentation of javax.sql.RowSetEvent interface (available in the JDBC 3.0 specification) for more details on how to register and handle standard RowSet events generated by compliant implementations.** |
| **Setters for a RowSet object's command** | **Provides a complete set of setter methods for setting RowSet command parameters.** |
| **Streams** | **Provides fields for storing of stream instances in addition to providing a set of constants for stream type designation.** |

* **3.2 Connected RowSet Requirements**  
  **The JdbcRowSet describes a RowSet object that must always be connected to the originating data source. Implementations of the JdbcRowSet should ensure that this connection is provided solely by a JDBC driver. Furthermore, RowSet objects that are implementations of the JdbcRowSet interface and are therefore operating in a connected environment do not use the SyncFactory to obtain a RowSetReader object or a RowSetWriter object. They can safely rely on the JDBC driver to supply their needs by virtue of the presence of an underlying updatable and scrollable ResultSet implementation.**
* **3.3 Disconnected RowSet Requirements**  
  **A disconnected RowSet object, such as a CachedRowSet object, should delegate connection management to a SyncProvider object provided by the SyncFactory. To ensure fully disconnected semantics, all disconnected RowSet objects must ensure that the original connection made to the data source to populate the RowSet object is closed to permit the garbage collector to recover and release resources. The SyncProvider object ensures that the critical JDBC properties are maintained in order to re-establish a connection to the data source when a synchronization is required. A disconnected RowSet object should therefore ensure that no extraneous references remain on the Connection object.**
* **3.4 Role of RowSetMetaDataImpl**  
  **The RowsetMetaDataImpl class is a utility class that provides an implementation of the** [**RowSetMetaData**](http://docs.google.com/RowSetMetaData.html) **interface, supplying standard setter method implementations for metadata for both connected and disconnected RowSet objects. All implementations are free to use this standard implementation but are not required to do so.**
* **3.5 RowSetWarning Class**  
  **The RowSetWarning class provides warnings that can be set on RowSet implementations. Similar to** [**SQLWarning**](http://docs.google.com/java/sql/SQLWarning.html) **objects, RowSetWarning objects are silently chained to the object whose method caused the warning to be thrown. All RowSet implementations should ensure that this chaining occurs if a warning is generated and also ensure that the warnings are available via the getRowSetWarnings method defined in either the JdbcRowSet interface or the CachedRowSet interface. After a warning has been retrieved with one of the getRowSetWarnings methods, the RowSetWarning method getNextWarning can be called on it to retrieve any warnings that might be chained on it. If a warning is returned, getNextWarning can be called on it, and so on until there are no more warnings.**
* **3.6 The Joinable Interface**  
  **The Joinable interface provides both connected and disconnected RowSet objects with the capability to be added to a JoinRowSet object in an SQL JOIN operation. A RowSet object that has implemented the Joinable interface can set a match column, retrieve a match column, or unset a match column. A JoinRowSet object can then use the RowSet object's match column as a basis for adding the RowSet object.**

### **4.0 Related Specifications**

* [**JDBC 3.0 Specification**](http://java.sun.com/products/jdbc)
* [**XML Schema**](http://www.w3.org/XML/Schema)
* [**SyncML**](http://www.syncml.org)

### **5.0 Related Documentation**

* [**JDBC RowSet Tutorial**](http://developer.java.sun.com/developer/Books/JDBCTutorial/chapter5.html)

| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | **Class** | [**Use**](http://docs.google.com/package-use.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV PACKAGE**](http://docs.google.com/javax/sql/package-summary.html)[**NEXT PACKAGE**](http://docs.google.com/javax/sql/rowset/serial/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sql/rowset/package-summary.html)[**NO FRAMES**](http://docs.google.com/package-summary.html)[**All Classes**](http://docs.google.com/allclasses-noframe.html) |

[**Submit a bug or feature**](http://bugs.sun.com/services/bugreport/index.jsp)

**For further API reference and developer documentation, see** [**Java SE Developer Documentation**](http://docs.google.com/webnotes/devdocs-vs-specs.html)**. That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.**

**Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to** [**license terms**](http://docs.google.com/legal/license.html)**. Also see the** [**documentation redistribution policy**](http://java.sun.com/docs/redist.html)**.**