ABOUT PRASAD SAYA

javax.sql.rowset.JdbcRowSet Example

Interface and its basic usage. This class is defined in the javax.sql.rowset package. JabbcRowSet (and Rowset extends Resultset Jisa a connected rowset. It is a connected rowset. It is a wrapper around a Resultset object; an enhanced Resultset object. It maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It can be used to make a Resultset It can be used to make a Resultset	⚠ Posted by: Prasad Saya 🖿 in JdbcRowSet, sql 🕚 February 3rd, 2015 👲 0 📀 340 Views
interface and its basic usage. This class is defined in the javax.sql.rowset javax.sql.rowset javax.sql.rowset javax.sql.rowset interface extends Rowset (and Rowset extends Resultset). A Jdbc rowset: • It is a connected rowset. • It is a connected rowset. • It is a wrapper around a Resultset object. It maintains a connection to its data source, just as a Resultset object. It maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. • It has a set of properties and a listener notification mechanism that make it a JavaBeans component. • It can be used to make a Resultset	This article introduces the
javax.sql.rowset jackage. jabcRowset interface extends Rowset (and Rowset extends Resultset). A Jabc rowset: • It is a connected rowset. • It is a wrapper around a Resultset object; an enhanced Resultset object; the maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jabc rowset can, for example, be a component in a Java Swing application. • It can be used to make a Resultset other than a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a Resultset	JdbcRowSet
javax.sql.rowset jackage. jabcRowset interface extends Rowset (and Rowset extends Resultset). A Jabc rowset: • It is a connected rowset. • It is a wrapper around a Resultset object; an enhanced Resultset object; the maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jabc rowset can, for example, be a component in a Java Swing application. • It can be used to make a Resultset other than a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a Resultset	interface and its basic usage. This class is defined in the
radbcRowSet (and RowSet (and RowSet (and RowSet extends ResultSet). A Jabc rowSet: • It is a connected rowSet. • It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet • It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	
radbcRowSet (and RowSet (and RowSet (and RowSet extends ResultSet). A Jabc rowSet: • It is a connected rowSet. • It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet • It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	
interface extends Rowset (and Rowset extends Resultset). A Jobc rowset: It is a connected rowset. It is a wrapper around a Resultset object; an enhanced Resultset object. It maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jobc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a Resultset	
Rowset (and Rowset extends ResultSet). A Jdbc rowset: It is a connected rowset. It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	JdbcRowSet
(and Rowset extends ResultSet). A Jdbc rowset: • It is a connected rowset. • It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. • It has a set of properties and a listener notification mechanism that make it a JavaBeans component. • It can be used to make a ResultSet	interface extends
Rowset extends ResultSet A Jdbc rowset: It is a connected rowset. It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	RowSet
Rowset extends ResultSet A Jdbc rowset: It is a connected rowset. It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	
extends Resultset A Jdbc rowset: It is a connected rowset. It is a wrapper around a Resultset object; an enhanced Resultset object. It maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a Resultset	(and
ResultSet). A Jdbc rowset: It is a connected rowset. It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	Rowset
A Jdbc rowset: It is a connected rowset. It is a wrapper around a Resultset object; an enhanced Resultset object. It maintains a connection to its data source, just as a Resultset object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	extends
It is a connected rowset. It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	ResultSet
It is a connected rowset. It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	
 It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet).
It is a wrapper around a ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	A Jdbc rowset:
ResultSet object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	It is a connected rowset.
object; an enhanced ResultSet object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	It is a wrapper around a
object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	ResultSet
object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	abjects an anhanced
object. It maintains a connection to its data source, just as a ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	
ResultSet object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	
object does. As a consequence, a Jdbc rowset can, for example, be a component in a Java Swing application. It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet	object. It maintains a connection to its data source, just as a
 It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet 	ResultSet
 It has a set of properties and a listener notification mechanism that make it a JavaBeans component. It can be used to make a ResultSet 	
It can be used to make a ResultSet	
ResultSet	
object scrollable and updateable when it does not otherwise have those capabilities.	resultset
Tagain and an apparatual filler is does not esterned have alone deposition	object scrollable and updateable when it does not otherwise have those capabilities.

1. JDBC Rowset

1.1. Connected Rowset

A rowset object may make a connection with a data source and maintain that connection throughout its life cycle, so it is called a connected rowset.

1.2. JavaBeans Properties

1.2. Javabeans Froperties
The
RowSet
interface provides a set of JavaBeans properties. This allows a
RowSet
instance to be configured to connect to a JDBC data source and read data from the data source:
setUrl()

,

```
setUserName()
setDataSourceName()
setQueryTimeOut()
setReadOnly()
setTransactionIsolation()
setCommand()
, ... and corresponding getter methods.
A group of setter methods (
setInt()
setByte()
setString()
, ...) provide a way to pass input parameters to a rowset's command property.
For example:
         JdbcRowSetImpl jrs = new JdbcRowSetImpl();
jrs.setCommand("SELECT * FROM books WHERE author = ?");
jrs.setURL("jdbc:myDriver:myAttribute"); // set method to connect to datasource (configure)
jrs.setUsername("myuser");
jrs.setPassword("mypwd");
jrs.setString(1, "Mark Twain"); // set method to pass input parameter
jrs.execute(); // fills this rowset object with data
1.3. JavaBeans Notification Mechanism
Rowset objects use the JavaBeans event model.
RowSetListener
is an interface that is implemented by a component that wants to be notified when a significant event happens in the life of a
object. A component becomes a listener by being registered with a
RowSet
object via the method
RowSet.addRowSetListener()
There are three events trigger notifications (and handled by listener methods):
• Cursor movement:
    cursorMoved(RowSetEvent)
• Update, insert or delete of a row:
     rowChanged(RowSetEvent)
• Change to the entire rowset content:
    rowSetChanged(RowSetEvent)
```

1.4. Creating a JDBC Rowset

There are four ways to create a

JdbcRowSet

```
JdbcRowSet
```

object needs to connect to database and then be populated with data.

1.4.1. Reference implementation default constructor

```
1    JdbcRowSet jdbcRs = new JdbcRowSetImpl(); // create rowset object
2    jdbcRs.setCommand("select * from BOOKS"); // set properties, and
3    jdbcRs.setUrl(url); // connect to database
4    jdbcRs.setUserName(usr);
5    jdbcRs.setPassword(pwd);
6    jdbcRs.execute(); // populate with data
```

1.4.2. Constructor that takes a Connection object

```
JdbcRowSet jdbcRs = JdbcRowSetImpl(conn); // conn is the java.sql.Connection object
jdbcRs.setCommand("select * from BOOKS");
jdbcRs.execute();
```

1.4.3. Constructor that takes a ResultSet object

The

```
ResultSet
```

must be created as updateable and scrollable; otherwise the Jdbc rowset will not be updateable as well.

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
ResultSet rs = stmt.executeQuery("select * from BOOKS");
JdbcRowSet jrs = new JdbcRowSetImpl(rs);
```

1.4.4. Using an instance of RowSetFactory

```
1 RowSetFactory rsf = RowSetProvider.newFactory();
2 JdbcRowSet jrs = rsf.createJdbcRowset();
3 // Set properties, connect to database and populate the rowset with data ...
```

The example program in this article uses a

```
RowSetFactory
```

to create a jdbc rowset.

1.5. Using JdbcRowSet objects

· Rowset can use all of the cursor movement methods defined in the

ResultSet

interface:

absolute(int row)

previous()

relative(int rows)

• Rowset is updated (insert, update and delete) the same way data is updated in a

ResultSet

object.

2. The Example

The example program performs Create, Read, Update and Delete operations on a database table using the

JDBCRowset

- Create the JDBC rowset, configure it and connect to the database.
- Read all rows from the database table and populate the rowset.
- Query all rows from rowset.
- Insert a row into the rowset.
- Update a row in the rowset.

• Create the JDBC rowset, configure it and confiect to the database.

· Delete a row from the rowset.

The example uses MySQL version 5.5.20 server database. The details to create the example database and data are shown below.

2.1. Database SQL Scripts

The following MySQL SQL commands can be used to create the example database, table and insert some data into the table. The

```
mysql
```

command-line tool can be used.

2.1.1. Create database, verify and use it

```
1 CREATE DATABASE example_db;
2 SHOW DATABASES;
3 USE example_db;
```

2.1.2. Create table and verify it

```
1 CREATE TABLE books_table (title VARCHAR(40), author VARCHAR(40), PRIMARY KEY (title));
2 DESCRIBE books_table;
```

2.1.3. Insert data into the table

```
INSERT INTO books_table VALUES ('The Mysterious Affair at Styles', 'Agatha Christie');
INSERT INTO books_table VALUES ('The Count of Monte Cristo', 'Alexandre Dumas');
INSERT INTO books_table VALUES ('A Study in Scarlet', 'Arthur Conan Doyle');
INSERT INTO books_table VALUES ('Doctor Zhivago', 'Boris Pasternak');
INSERT INTO books_table VALUES ('Animal Farm', 'George Orwell');
```

2.1.4. Query the table data

```
1 | SELECT * FROM books_table;
```

The output from the query should be as follows:

```
01 | mysql> SELECT * FROM books_table;
02
                                        | author
94
    A Study in Scarlet
Animal Farm
                                          Arthur Conan Doyle
05
                                          George Orwell
     Doctor Zhivago
97
                                          Boris Pasternak
     The Count of Monte Cristo
80
                                          Alexandre Dumas
    The Mysterious Affair at Styles
                                        Agatha Christie
10
11 5 rows in set (0.00 sec)
```

2.2. The Example Program Code

JDBCRowsetExample.java

```
import javax.sql.rowset.RowSetProvider;
import javax.sql.rowset.RowSetFactory;
import javax.sql.rowset.JdbcRowSet;
002
003
        import java.sql.SQLException;
005
006
        public class JDBCRowsetExample {
007
008
               private int insertedRowNo;
009
              private final static String DB_URL = "jdbc:mysql://localhost:3306/example_db";
private final static String USR = "root";
private final static String PWD = "root";
private final static String BOOKS_TABLE = "books_table";
private final static String TITLE = "title";
private final static String AUTHOR = "author";
010
011
012
013
014
015
016
               private final static String INSERT_ROW_TITLE = "Lady Chatterley's Lover";
private final static String INSERT_ROW_AUTHOR = "D H Lawrence";
private final static String UPDATE_ROW_AUTHOR = "D H LAWRENCE";
017
018
019
020
021
               public JDBCRowsetExample() {
022
023
024
               public static void main(String [] args)
025
                             throws Exception {
026
                      JDBCRowsetExample pgm = new JDBCRowsetExample();
027
028
029
                      JdbcRowSet jrs = pgm.getJDBCRowset();
030
                      pgm.loadAllRows(jrs);
031
                      pgm.printAllRows(jrs);
032
033
                      pgm.insertRow(jrs);
                      pgm.updateRow(jrs);
pgm.deleteRow(jrs);
034
035
036
                      jrs.close();
037
```

```
039
                 System.out.println("- Close."):
040
041
042
            private JdbcRowSet getJDBCRowset()
043
                       throws SQLException {
044
045
                 System.out.println("- Configure JDBC Rowset and connect to database: " + DB URL);
046
                 RowSetFactory rsFactory = RowSetProvider.newFactory();
JdbcRowSet jRowset = rsFactory.createJdbcRowSet();
047
048
049
                  jRowset.setUsername(USR):
050
051
                  jRowset.setPassword(PWD);
052
                  jRowset.setUrl(DB_URL);
                  jRowset.setReadOnly(false); // make rowset updateable
053
054
055
                 return jRowset;
056
058
059
            private void loadAllRows(JdbcRowSet jrs)
                       throws SQLException {
060
061
                 // populate the rowset with table rows
062
                 System.out.println("- Load (initial) all rows from database table: " + BOOKS_TABLE);
String sql = "SELECT * FROM " + BOOKS_TABLE;
jrs.setCommand(sql);
063
964
065
066
                 jrs.execute();
067
068
                 System.out.println("Total rows in table: " + getRowCount(jrs));
069
070
071
            private int getRowCount(JdbcRowSet jrs)
072
                       throws SQLException {
073
074
                 jrs.last();
075
                  return jrs.getRow();
076
077
078
            private void printAllRows(JdbcRowSet jrs)
079
                       throws SQLException {
080
                 System.out.println("- Print all rows:");
081
082
083
                 jrs.beforeFirst();
084
085
                 while (irs.next()) {
086
                       int rowNo = jrs.getRow();
String s1 = jrs.getString(TITLE);
String s2 = jrs.getString(AUTHOR);
System.out.println(rowNo + ": " + s1 + ", " + s2);
087
088
089
999
091
                 }
092
            }
093
094
            private void insertRow(JdbcRowSet jrs)
095
                       throws SQLException {
096
097
                 System.out.println("- Insert row: ");
098
                 jrs.moveToInsertRow();
jrs.updateString(TITLE, INSERT_ROW_TITLE);
jrs.updateString(AUTHOR, INSERT_ROW_AUTHOR);
099
100
101
102
                  jrs.insertRow();
103
                 insertedRowNo = jrs.getRow(); // Note: this is an instance variable
String s1 = jrs.getString(TITLE);
String s2 = jrs.getString(AUTHOR);
System.out.println(insertedRowNo + ": " + jrs.getString(TITLE) + ", " + jrs.getString(AUTHOR));
System.out.println("Total rows in table: " + getRowCount(jrs));
104
105
106
107
108
109
110
            private void updateRow(JdbcRowSet jrs)
111
                       throws SQLException {
112
113
114
                 System.out.println("- Update row " + insertedRowNo);
115
                  irs.absolute(insertedRowNo);
116
                 String s1 = jrs.getString(TITLE);
String s2 = jrs.getString(AUTHOR);
System.out.println("Row (before update): " + s1 + ", " + s2);
117
118
119
120
121
                 jrs.updateString("AUTHOR", UPDATE_ROW_AUTHOR);
jrs.updateRow();
122
123
                 s1 = jrs.getString(TITLE);
s2 = jrs.getString(AUTHOR);
System.out.println("Row (after update): " + s1 + ", " + s2);
124
125
126
127
128
129
            private void deleteRow(JdbcRowSet jrs)
130
                       throws SQLException {
131
                 jrs.absolute(insertedKowNo),
String s1 = jrs.getString(TITLE);
String s2 = jrs.getString(AUTHOR);
String s0 = println("- Delete row " + insertedRowNo + ": " + s1 + ", " + s2);
132
133
134
135
136
137
                 irs.deleteRow();
138
                 System.out.println("Total rows in table: " + getRowCount(jrs));
139
140
            }
141 }
```

2.3. The Output

```
- Configure JDBC Rowset and connect to database: jdbc:mysql://localhost:3306/example_db - Load (initial) all rows from database table: books_table

Total rows in table: 5
       Print all rows:

1: A Study in Scarlet, Arthur Conan Doyle
2: Animal Farm, George Orwell
3: Doctor Zhivago, Boris Pasternak
4: The Count of Monte Cristo, Alexandre Dumas
5: The Mysterious Affair at Styles, Agatha Christie
        - Insert row:
10
        6: Lady Chatterley's Lover, D H Lawrence
       Total rows in table: 6
- Update row 6
13
       Row (before update): Lady Chatterley's Lover, D H Lawrence
Row (after update): Lady Chatterley's Lover, D H LAWRENCE
- Delete row 6: Lady Chatterley's Lover, D H LAWRENCE
Total rows in table: 5
18 - Close.
```

From the output:

The program inserts a row with title "Lady Chatterley's Lover" and author "D H Lawrence". Next, it updates the same row's author to "D H LAWRENCE". Finally, it deletes the inserted (and updated) row.

3. Download Java Source Code

This was an example of

javax.sql.rowset.JdbcRowSet Example

Download

You can download the full source code of this example here: JDBCRowsetExample.zip



(No Ratings Yet) Start the discussion 340 Views Tweet it!

Do you want to know how to develop your skillset to become a Java Rockstar?

> Subscribe to our newsletter to start Rocking right now! To get you started we give you our best selling eBooks for FREE!

- 1. JPA Mini Book
- 2. JVM Troubleshooting Guide
- 3. JUnit Tutorial for Unit Testing
- 4. Java Annotations Tutorial
- 5. Java Interview Questions
- 6. Spring Interview Questions
- 7. Android UI Design

and many more

Email address:

Your email address

Receive Java & Developer job alerts in your Area

Sian up

LIKE THIS ARTICLE? READ MORE FROM JAVA CODE GEEKS

Leave a Reply



Start the discussion...