

## Database Access with JDBC

Core Servlets & JSP book: [www.coreservlets.com](http://www.coreservlets.com)  
 More Servlets & JSP book: [www.moreservlets.com](http://www.moreservlets.com)  
 Servlet and JSP Training Courses: [courses.coreservlets.com](http://courses.coreservlets.com)

Slides © Marty Hall, <http://www.moreservlets.com>, book © Sun Microsystems Press

## Overview

- Overview of JDBC technology
- JDBC drivers
- Seven basic steps in using JDBC
- Retrieving data from a ResultSet
- Using prepared and callable statements
- Handling SQL exceptions
- Submitting multiple statements as a transaction

www.moreservlets.com

## JDBC Introduction

- **JDBC provides a standard library for accessing relational databases**
  - API standardizes
    - Way to establish connection to database
    - Approach to initiating queries
    - Method to create stored (parameterized) queries
    - The data structure of query result (table)
      - Determining the number of columns
      - Looking up metadata, etc.
  - API does not standardize SQL syntax
    - JDBC is not embedded SQL
  - JDBC classes are in the java.sql package
- **Note: JDBC is not officially an acronym; unofficially, "Java DataBase Connectivity" is commonly used**

www.moreservlets.com

## On-line Resources

- **Sun's JDBC Site**
  - <http://java.sun.com/products/jdbc/>
- **JDBC Tutorial**
  - <http://java.sun.com/docs/books/tutorial/jdbc/>
- **List of Available JDBC Drivers**
  - <http://industry.java.sun.com/products/jdbc/drivers/>
- **API for java.sql**
  - <http://java.sun.com/j2se/1.4/docs/api/java/sql/package-summary.html>

www.moreservlets.com

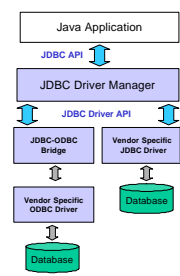
## Oracle On-line Resources

- **JDBC Road Map**
  - <http://technet.oracle.com/tech/java/jroadmap/index2.htm?Info&jdbc/listing.htm>
- **SQLJ & JDBC Basic Samples**
  - [http://technet.oracle.com/tech/java/sqlj\\_jdbc/index2.htm?Code&files/basic/basic.htm](http://technet.oracle.com/tech/java/sqlj_jdbc/index2.htm?Code&files/basic/basic.htm)
- **JDBC Drivers**
  - [http://technet.oracle.com/software/tech/java/sqlj\\_jdbc/htdocs/listing.htm](http://technet.oracle.com/software/tech/java/sqlj_jdbc/htdocs/listing.htm)
  - Requires free registration
- **Certification**
  - <http://technet.oracle.com/training/certification/>

www.moreservlets.com

## JDBC Drivers

- **JDBC consists of two parts:**
  - JDBC API, a purely Java-based API
  - JDBC Driver Manager, which communicates with vendor-specific drivers that perform the real communication with the database.
    - Point: translation to vendor format is performed on the client
      - No changes needed to server
      - Driver (translator) needed on client



www.moreservlets.com

## JDBC Data Types

JDBC Type	Java Type	JDBC Type	Java Type
BIT	boolean	NUMERIC	BigDecimal
TINYINT	byte	DECIMAL	BigDecimal
SMALLINT	short	DATE	java.sql.Date
INTEGER	int	TIME	java.sql.Timestamp
BIGINT	long	TIMESTAMP	java.sql.Timestamp
REAL	float	CLOB	Clob
FLOAT	double	BLOB	Blob
DOUBLE	double	ARRAY	Array
BINARY	byte[]	DISTINCT	mapping of underlying type
VARBINARY	byte[]	STRUCT	Struct
LONGVARBINARY	byte[]	REF	Ref
CHAR	String	JAVA_OBJECT	underlying Java class
VARCHAR	String		
LONGVARCHAR	String		

\*SQL3 data type supported in JDBC 2.0

## Seven Basic Steps in Using JDBC

- Load the driver
- Define the Connection URL
- Establish the Connection
- Create a Statement object
- Execute a query
- Process the results
- Close the connection

## JDBC: Details of Process

- Load the driver

```
try {
    Class.forName("connect.microsoft.MicrosoftDriver");
    Class.forName("oracle.jdbc.driver.OracleDriver");
} catch (ClassNotFoundException cnfe) {
    System.out.println("Error loading driver: " + cnfe);
}
```

- Define the Connection URL

```
String host = "dbhost.yourcompany.com";
String dbName = "someName";
int port = 1234;
String oracleURL = "jdbc:oracle:thin:@" + host +
    ":" + port + ":" + dbName;
String sybaseURL = "jdbc:sybase:Tds:" + host +
    ":" + port + ":" +
    "?SERVERNAME=" + dbName;
```

## JDBC: Details of Process (Continued)

- Establish the Connection

```
String username = "jay_debese";
String password = "secret";
Connection connection =
    DriverManager.getConnection(oracleURL,
        username,
        password);
```

- Optionally, look up information about the database

```
DatabaseMetaData dbMetaData =
    connection.getMetaData();
String productName =
    dbMetaData.getDatabaseProductName();
System.out.println("Database: " + productName);
String productVersion =
    dbMetaData.getDatabaseProductVersion();
System.out.println("Version: " + productVersion);
```

## JDBC: Details of Process (Continued)

- Create a Statement

```
Statement statement =
    connection.createStatement();
```

- Execute a Query

```
String query =
    "SELECT col1, col2, col3 FROM sometable";
ResultSet resultSet =
    statement.executeQuery(query);
```

- To modify the database, use executeUpdate, supplying a string that uses UPDATE, INSERT, or DELETE
- Use setQueryTimeout to specify a maximum delay to wait for results

## JDBC: Details of Process (Continued)

- Process the Result

```
while(resultSet.next()) {
    System.out.println(resultSet.getString(1) + " " +
        resultSet.getString(2) + " " +
        resultSet.getString(3));
}
```

- First column has index 1, not 0
- ResultSet provides various getXxx methods that take a colu index *or* column name and returns the data
- You can also access result meta data (column names, etc.)

- Close the Connection

```
connection.close();
```

- Since opening a connection is expensive, postpone this step if additional database operations are expected

### The Microsoft Access Northwind Database

- Database that comes preinstalled with Microsoft Office

Last Name	First Name	Title	Title Of Courtesy	Birth Date
Adams	Nancy	Sales Representative	Ms.	09/08/1962
Peacock	Janet	Sales Representative	Ms.	08/30/1957
Chavez	Margaret	Sales Representative	Ms.	01/30/1958
Dee	Michael	Sales Representative	Ms.	02/18/1960
King	Ruben	Sales Representative	Ms.	09/16/1957
Callahan	Laura	Inside Sales Coordinator	Ms.	05-16-1955
Edwards	Aruno	Sales Representative	Ms.	02-18-1960

www.moreservlets.com

### Using Microsoft Access via ODBC

- Click Start, Settings, Control Panel, Administrative Tools, Data Sources, System DSN, and select Add

www.moreservlets.com

### Using Microsoft Access via ODBC (Continued)

- Select Microsoft Access Driver, Finish, type a name under Data Source Name, and hit Select

www.moreservlets.com

### Using Microsoft Access via ODBC (Continued)

- Navigate to the Samples directory of MS Office, select Northwind.mdb, hit OK, then hit OK in following two windows

www.moreservlets.com

### Using Microsoft Access via ODBC (Continued)

- Use `sun.jdbc.odbc.JdbcOdbcDriver` as the class name of the JDBC driver.
  - `Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");`
- Use `"jdbc:odbc:Northwind"` as the database address, and use empty strings for the username and password.
  - `Connection connection = DriverManager.getConnection("jdbc:odbc:Northwind", "", "");`

www.moreservlets.com

### Simple Standalone Northwind Test

```

package coreservlets;

import java.sql.*;

public class NorthwindTest {
    public static void main(String[] args) {
        String driver =
            "sun.jdbc.odbc.JdbcOdbcDriver";
        String url = "jdbc:odbc:Northwind";
        String username = "";
        String password = "";
        showEmployeeTable(driver, url,
            username, password);
    }
}
    
```

www.moreservlets.com

## Simple Standalone Northwind Test (Continued)

```
public static void showEmployeeTable(String driver,
                                    String url,
                                    String username,
                                    String password) {
    try {
        // Load database driver if not already loaded.
        Class.forName(driver);
        // Establish network connection to database.
        Connection connection =
            DriverManager.getConnection(url,
                                       username, password);
        System.out.println("Employees\n" +
                           "=====");
        Statement statement = connection.createStatement();
        String query =
            "SELECT firstname, lastname FROM employees";
        // Send query to database and store results.
        ResultSet resultSet = statement.executeQuery(query);
```

18

JDBC

www.moreservlets.com

## Simple Standalone Northwind Test (Continued)

```
        // Print results.
        while(resultSet.next()) {
            // First name
            System.out.print(resultSet.getString(1) + " ");
            // Last name
            System.out.println(resultSet.getString(2));
        }
        catch(ClassNotFoundException cnfe) {
            System.err.println("Error loading driver: " + cnfe);
        }
        catch(SQLException sqle) {
            System.err.println("Error connecting: " + sqle);
        }
    }
}
```

19

JDBC

www.moreservlets.com

## Simple Standalone Northwind Test: Results

Prompt> java coreservlets.NorthwindTest

```
Employees
=====
Nancy Davolio
Andrew Fuller
Janet Leverling
Margaret Peacock
Steven Buchanan
Michael Suyama
Robert King
Laura Callahan
Anne Dodsworth
```

21

JDBC

www.moreservlets.com

## Using MetaData

- **System-wide data**
  - connection.getMetaData().getDatabaseProductName()
  - connection.getMetaData().getDatabaseProductVersion()
- **Table-specific data**
  - resultSet.getMetaData().getColumnCount()
    - When using the result, remember that the index starts at 1, not 0
  - resultSet.getMetaData().getColumnName()

22

JDBC

www.moreservlets.com

## Using MetaData: Example

```
public class NorthwindServlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                     HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        ... out.println(docType + ...);
        String driver = "sun.jdbc.odbc.JdbcOdbcDriver";
        String url = "jdbc:odbc:Northwind";
        String username = "";
        String password = "";
        String tableName = request.getParameter("tableName");
        if ((tableName == null) || (tableName.equals("")) {
            tableName = "employees";
        }
        showTable(driver, url, username, password,
                 tableName, out);
        out.println("</CENTER></BODY></HTML>");
    }
}
```

23

JDBC

www.moreservlets.com

## Using MetaData: Example (Continued)

```
private void showTable(String driver,
                      String url,
                      String username,
                      String password,
                      String tableName,
                      PrintWriter out) {
    try {
        Class.forName(driver);
        Connection connection =
            DriverManager.getConnection(url, username, password);
        DatabaseMetaData dbMetaData = connection.getMetaData();
        out.println("<UL>");
        String productName =
            dbMetaData.getDatabaseProductName();
        out.println(" <LI><B>Database:</B> " +
                   productName);
        String productVersion =
            dbMetaData.getDatabaseProductVersion();
        out.println(" <LI><B>Version:</B> " +
                   productVersion +
                   "\n</UL>");
    }
}
```

24

JDBC

www.moreservlets.com

### Using MetaData: Example (Continued)

```

Statement statement = connection.createStatement();
String query =
    "SELECT * FROM " + tableName;
ResultSet resultSet = statement.executeQuery(query);
out.println("<TABLE BORDER=1>");
ResultSetMetaData resultsMetaData =
    resultSet.getMetaData();
int columnCount = resultsMetaData.getColumnCount();
out.println("<TR>");
for(int i=1; i<columnCount+1; i++) {
    out.print("<TH> " + resultsMetaData.getColumnLabel(i));
}
out.println();
while(resultSet.next()) {
    out.println("<TR>");
    for(int i=1; i<columnCount+1; i++) {
        out.print("<TD>" + resultSet.getString(i));
    }
    out.println();
}
out.println("</TABLE>");
    
```

www.moreservlets.com

### Using MetaData: Results

### Using Statement

- Overview**
  - Through the Statement object, SQL statements are sent to the database.
  - Three types of statement objects are available:
    - Statement**
      - For executing a simple SQL statement
    - PreparedStatement**
      - For executing a precompiled SQL statement passing in parameters
    - CallableStatement**
      - For executing a database stored procedure

www.moreservlets.com

### Useful Statement Methods

- executeQuery**
  - Executes the SQL query and returns the data in a table (ResultSet)
  - The resulting table may be empty but never null

```

ResultSet results =
    statement.executeQuery("SELECT a, b FROM table");
    
```
- executeUpdate**
  - Used to execute for INSERT, UPDATE, or DELETE SQL statements
  - The return is the number of rows that were affected in the database
  - Supports Data Definition Language (DDL) statements CREATE TABLE, DROP TABLE and ALTER TABLE

```

int rows =
    statement.executeUpdate("DELETE FROM EMPLOYEES" +
        "WHERE STATUS=0");
    
```

www.moreservlets.com

### Useful Statement Methods (Continued)

- execute**
  - Generic method for executing stored procedures and prepared statements
  - Rarely used (for multiple return result sets)
  - The statement execution may or may not return a ResultSet (use statement.getResultSet). If the return value is true, two or more result sets were produced
- getMaxRows/setMaxRows**
  - Determines the maximum number of rows a ResultSet may contain
  - Unless explicitly set, the number of rows is unlimited (return value of 0)
- getQueryTimeout/setQueryTimeout**
  - Specifies the amount of a time a driver will wait for a STATEMENT to complete before throwing a SQLException

www.moreservlets.com

### Prepared Statements (Precompiled Queries)

- Idea**
  - If you are going to execute similar SQL statements multiple times, using "prepared" (parameterized) statements can be more efficient
  - Create a statement in standard form that is sent to the database for compilation before actually being used
  - Each time you use it, you simply replace some of the marked parameters using the setXxx methods
- As PreparedStatement inherits from Statement the corresponding execute methods have no parameters**
  - execute()
  - executeQuery()
  - executeUpdate()

www.moreservlets.com

## Prepared Statement, Example

```

Connection connection =
    DriverManager.getConnection(url, user,
        password);
PreparedStatement statement =
    connection.prepareStatement("UPDATE employees "+
        "SET salary = ? " +
        "WHERE id = ?");

int[] newSalaries = getSalaries();
int[] employeeIDs = getIDs();
for(int i=0; i<employeeIDs.length; i++) {
    statement.setInt(1, newSalaries[i]);
    statement.setInt(2, employeeIDs[i]);
    statement.executeUpdate();
}

```

31

JDBC

www.moreservlets.com

## Useful Prepared Statement Methods

- **setXxx**
  - Sets the indicated parameter (?) in the SQL statement to the value
- **clearParameters**
  - Clears all set parameter values in the statement
- **Handling Servlet Data**
  - Query data obtained from a user through an HTML form may have SQL or special characters that may require escape sequences
  - To handle the special characters, pass the string to the PreparedStatement **setString** method which will automatically escape the string as necessary

32

JDBC

www.moreservlets.com

## Transactions

- **Idea**
  - By default, after each SQL statement is executed the changes are **automatically committed** to the database
  - Turn auto-commit off to group two or more statements together into a transaction
 

```
connection.setAutoCommit(false)
```
  - Call **commit** to permanently record the changes to the database after executing a group of statements
  - Call **rollback** if an error occurs

33

JDBC

www.moreservlets.com

## Transactions: Example

```

Connection connection =
    DriverManager.getConnection(url, username, passwd);
connection.setAutoCommit(false);
try {
    statement.executeUpdate(...);
    statement.executeUpdate(...);

    connection.commit();
} catch (Exception e) {
    try {
        connection.rollback();
    } catch (SQLException sqle) {
        // report problem
    }
} finally {
    try {
        connection.close();
    } catch (SQLException sqle) { }
}

```

34

JDBC

www.moreservlets.com

## Useful Connection Methods (for Transactions)

- **getAutoCommit/setAutoCommit**
  - By default, a connection is set to auto-commit
  - Retrieves or sets the auto-commit mode
- **commit**
  - Force all changes since the last call to commit to become permanent
  - Any database locks currently held by this Connection object are released
- **rollback**
  - Drops all changes since the previous call to commit
  - Releases any database locks held by this Connection object

35

JDBC

www.moreservlets.com

## More JDBC Options

- **Stored procedures**
- **Changing buffer size**
- **Connection pooling**
- **JSP Standard Tag Library (JSTL) – custom tags to hide JDBC details**

36

JDBC

www.moreservlets.com

## Summary

- **You use the same Java syntax with all databases**
  - Translation to native format is done on the client via a JDBC driver
  - Standardized Java syntax does not equate to standardized SQL syntax
- **Steps in using JDBC**
  - Load the driver
  - Define the Connection URL
  - Establish the Connection
  - Create a Statement object
  - Execute a query
  - Process the results
  - Close the connection

37 JDBC [www.moreservlets.com](http://www.moreservlets.com)




## Questions?

Core Servlets & JSP book: [www.coreservlets.com](http://www.coreservlets.com)  
 More Servlets & JSP book: [www.moreservlets.com](http://www.moreservlets.com)  
 Servlet and JSP Training Courses: [courses.coreservlets.com](http://courses.coreservlets.com)

Slides © Marty Hall, <http://www.moreservlets.com>, book © Sun Microsystems Press

## More Information

- **Source code for examples**
  - <http://www.moreservlets.com>
- **More Servlets & JSP**
  - <http://www.moreservlets.com>
  - [Site includes info on servlet and JSP training courses](http://www.moreservlets.com)
- **Core Servlets & JSP**
  - Prequel to *More Servlets & JSP*
  - <http://www.coreservlets.com>
- **Servlet home page**
  - <http://java.sun.com/products/servlet/>
- **JavaServer Pages home page**
  - <http://java.sun.com/products/jsp/>



38 JDBC [www.moreservlets.com](http://www.moreservlets.com)