CSE 462 – Introduction to JDBC

Recitation Outline

- What is Java Database Connectivity (JDBC)?
  - JDBC Architecture
  - JDBC API
  - Using JDBC
- References

CSE 462 – Introduction to JDBC

What is JDBC?

- A Java API for querying and updating relational databases
- Relatively simple
- Standards driven (JCP)
- Promotes reusability of data access code
- Well-established (since JDK 1.1)

CSE 462 – Introduction to JDBC

JDBC Architecture

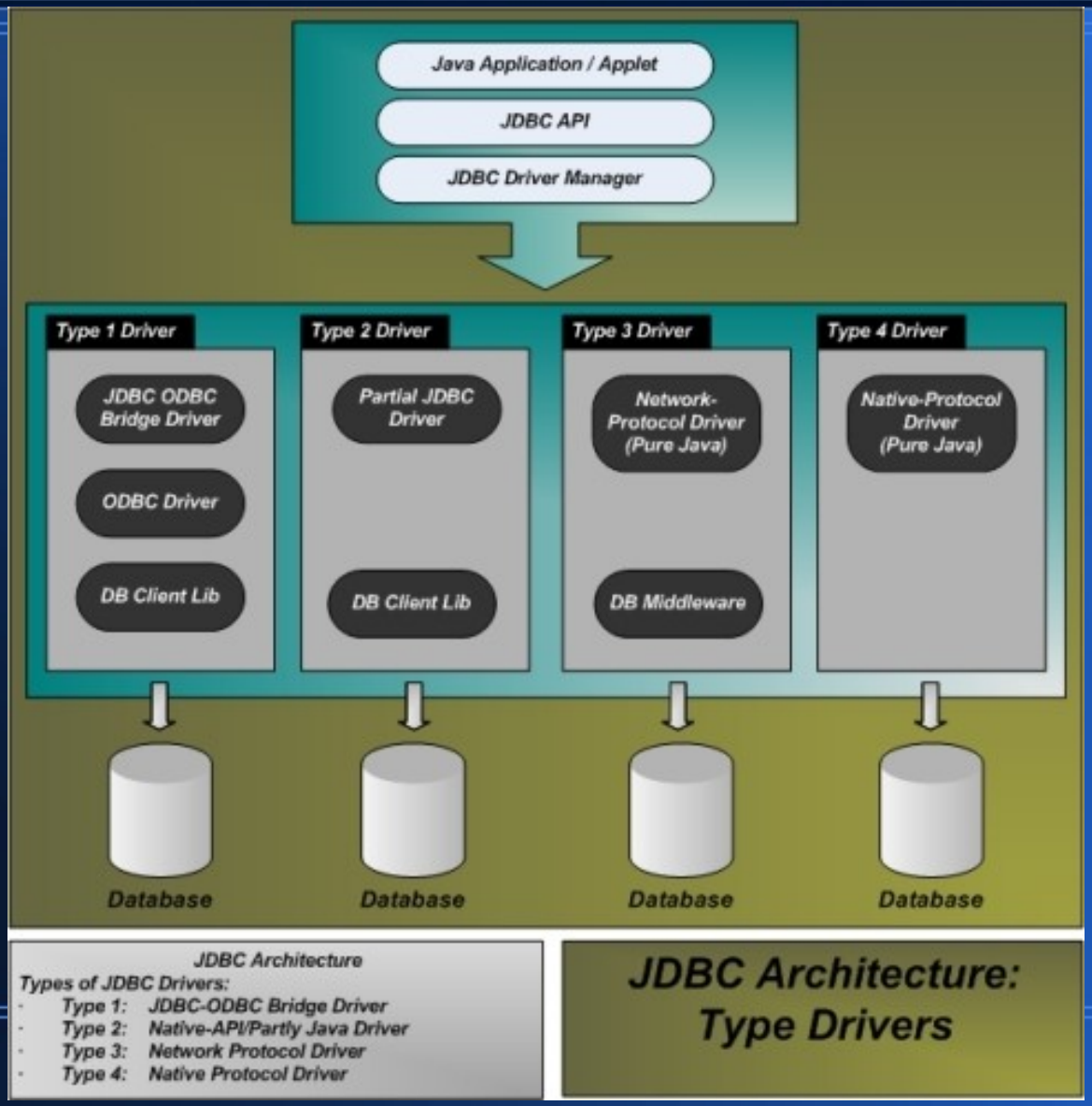
- Application (JDBC “client”)
  - Uses the JDBC API to access the data
- JDBC API
  - Provides programmatic access to the relational database
  - Communicates with the Driver Manager

CSE 462 – Introduction to JDBC

JDBC Architecture (continued)

- Driver Manager
  - Forwards application requests to the JDBC Driver
  - Forwards database responses to the application
- JDBC Driver
  - Database specific (e.g., Oracle JDBC driver)
  - Drivers come in FOUR types or implementations

CSE 462 – Introduction to JDBC



source: www.javadesign.info

CSE 462 – Introduction to JDBC

- JDBC API
  - Connection
    - Establishes a session with the database
    - Factory for JDBC Statement objects
  - DataSource
    - Factory for Connection objects
    - Preferred means of getting a Connection
  - DriverManager
    - Alternative way of getting a Connection

CSE 462 – Introduction to JDBC

JDBC API (continued)

- Statement
  - Executes simple SQL statements
- PreparedStatement (extends Statement)
  - A SQL statement is precompiled for multiple executions
  - Precompilation guarantees efficient execution
  - The SQL statement may be parameterized ('?' placeholders)
  - Each execution may bind different values to parameters

CSE 462 – Introduction to JDBC

JDBC API (continued)

- CallableStatement (extends PreparedStatement)
  - Executes stored procedures
- ResultSet
  - Represents a set of tuples
  - ResultSets are obtained by executing Statement objects

CSE 462 – Introduction to JDBC

JDBC API (continued)

- DatabaseMetaData
  - Comprehensive information about the database
- ResultSetMetaData
  - Information about types and properties of ResultSet columns

CSE 462 – Introduction to JDBC

Using JDBC: Getting a Connection (approach #1)

```
try {
    Class.forName("oracle.jdbc.driver.OracleDriver");
    String user   = "dlessa";
    String passwd = "my_password";
    String url    = "jdbc:oracle:thin:" + user + "/" + passwd +
        "@aos.acsu.buffalo.edu:1521/aos.buffalo.edu";
    Connection conn = DriverManager.getConnection(url);
}
catch (ClassNotFoundException e1) {
    System.out.println("Could not load the JDBC driver");
    e1.printStackTrace();
    System.exit(0);
}
catch (SQLException e2) {
    System.out.println("Could not connect to Oracle");
    e2.printStackTrace();
    System.exit(0);
}
```

CSE 462 – Introduction to JDBC

Using JDBC: Getting a Connection (approach #2)

```
try {
    OracleDataSource ds = new OracleDataSource();
    ds.setUser("dlessa");
    ds.setPassword("my_password");
    ds.setURL("jdbc:oracle:thin:@aos.acsu.buffalo.edu:1521/aos.buffalo.edu");
    Connection conn = ds.getConnection();
}
catch (SQLException e) {
    System.out.println("Could connect to the Oracle database");
    e.printStackTrace();
    System.exit(0);
}
```

CSE 462 – Introduction to JDBC

Using JDBC: Statement

```
Statement stmt = conn.createStatement();
try {
    String sql = "CREATE TABLE CITY ...";
    stmt.execute(sql);
}
catch (SQLException e) {
    System.out.println("Error creating CITY table");
    e.printStackTrace();
}
finally {
    stmt.close();
}
```

CSE 462 – Introduction to JDBC

Using JDBC: PreparedStatement

```
String sql = "INSERT INTO CITY(CITY_ID, CITY_NAME) values (?, ?)";
PreparedStatement pstmt = conn.prepareStatement(sql);
try {
    for ( int i = 0; i < 100; i++ ) {
        pstmt.setInt(1, i);
        pstmt.setString(2, "City #" + i);
        pstmt.executeUpdate();
    }
}
catch (SQLException e) {
    System.out.println("Error inserting CITY data");
    e.printStackTrace();
}
finally {
    pstmt.close();
}
```

CSE 462 – Introduction to JDBC

Using JDBC: ResultSet

```
String sql = "SELECT CITY_NAME FROM CITY";
ResultSet rs = stmt.executeQuery(sql);
try {
    System.out.println("All city names:");
    while (rs.next()) {
        System.out.println(rs.getString("CITY_NAME"));
    }
}
catch (SQLException e) {
    System.out.println("Error processing CITY data");
    e.printStackTrace();
}
finally {
    rs.close();
}
```

CSE 462 – Introduction to JDBC

Using JDBC: What does this code snippet do?

```
Statement s = con.createStatement();
ResultSet r = s.executeQuery("SELECT TABLE_NAME FROM USER_TABLES");
while (r.next()) {
    String tableName = r.getString(1);
    System.out.println(tableName);
    Statement rowCount = con.createStatement();
    System.out.println("records: ");
    rowCount.execute("SELECT COUNT(*) FROM " + tableName);
    print(rowCount.getResultSet());
    System.out.println("");
}
```

CSE 462 – Introduction to JDBC

References

Sun JDBC Tutorial

http://java.sun.com/docs/books/tutorial/jdbc/

JDBC Java Docs

http://java.sun.com/javase/6/docs/api/index.html?java/sql/package-summary.html

http://java.sun.com/javase/6/docs/api/index.html?javax/sql/package-summary.html

CSE IT Catalog: How to Use JDBC with Oracle

https://wiki.cse.buffalo.edu/services/content/how-use-jdbc-oracle