JDBC & ODBC!

Masashi Ito

Databases

- Used *everywhere*
  - Development (debugging)
  - General accounting
  - Web applications
    - HalfStats (http://vector.remotepoint.com/halfstats/)
      - Uses MySQL with a PHP3 web interface

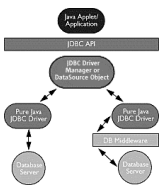
SQL (Structured Query Language)

- Pretty much a standard (ANSI)
  - Many SQL servers have special features that are not standard (Microsoft, mSQL, MySQL, etc)

JDBC (Java Database Connectivity)

- An API which enables SQL calls to be made from Java, and also translates SQL data types to Java data types
- Advantages:
  - *Very* portable (Java runs on practically all platforms, and the API allows access to many databases without recompilation/relink, provided you have the drivers)
- Disadvantages:
  - Performance
    - Java is slow
    - Going through a translation driver slows the program down. If you want speed, it might be better to use a native API

JDBC-How it works



JDBC-Code sample

```
import java.sql.*;

public class Sample
{
    public static void main(String args[])
    {
        String url = "jdbc:mySubprotocol:myDataSource"; // URL of the database
        Connection con;                                 // instatiation of a connection

        try
        {
            Class.forName("myDriver.Classname");        // attempt to load a JDBC driver
        }

        catch(java.lang.ClassNotFoundException e)       // catch the error if you get one
        {
            System.err.print("Class not found exception: ");
            System.err.println(e.getMessage());
        }
```

JDBD-Code sample (part 2)

```
try
{
    con = DriverManager.getConnection(url, "myLogin", "myPassword");

    // this string is the actual SQL statement
    String createNewTable = "CREATE TABLE BLAH " +
                            "((NAME VARCHAR(32), ROOM_NUM INTEGER, GPA FLOAT)";

    // initialize Statement object from Conenction
    Statement stmt = con.createStatement();
    stmt.executeUpdate(createNewTable);        // woohoo! we have a table

    // let's start inserting stuff
    stmt.executeUpdate("INSERT INTO NAME " + "values('Johnny', 100, 1.24)");
    stmt.executeUpdate("INSERT INTO NAME " + "values('Julie', 45, 4.00)");
    stmt.executeUpdate("INSERT INTO NAME " + "values('Joe', 50, 2.50)");

    // let's extract stuff
    ResultSet rs = stmt.executeQuery("select NAME, GPA from BLAH");
    while (rs.next())
    {
        String name = rs.getString("NAME");
        int roomnum = rs.getInt("ROOM_NUM");
        float gpa = rs.getFloat("GPA");

        System.out.println(name + " " + roomnum + " " + gpa);
    }
}
```

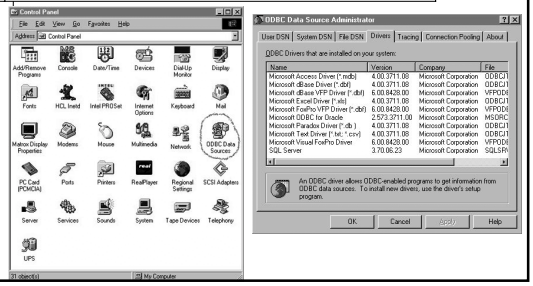
JDBC-Code sample (part 3)

```
        catch(SQLException ex)
        {
            System.err.println("SQL Exception: " + ex.getMessage());
        }
    }
}
```

ODBC (Open Database Connectivity)

- API that interfaces with various databases (SQL, Oracle, Access, dBASE, etc.)
- Interfaces through a driver
- Advantages (mostly the same as those of JDBC)
  - A single API for all the databases supported
  - No recompiling or relinking necessary when adding a new database type. Simply install the driver

You have ODBC (Windoze users)



JDBC $\Rightarrow$ ODBC Bridge

- The bridge is a JDBC driver which translates JDBC operations to ODBC operations
- There may not be a direct JDBC driver written for a particular database type of interest
- Obviously, there is greater overhead in using this bridge (but slower access is better than no access!!!)