

## Getting Started with Tomcat/NetBeans/JDBC

Bryan Duggan, NCI

### Introduction:

Tomcat is a combined web server and JSP/servlet host. It can serve up static HTML pages and dynamic pages generated using servlet or JSP technologies. Tomcat is free for commercial use and open source.

Netbeans

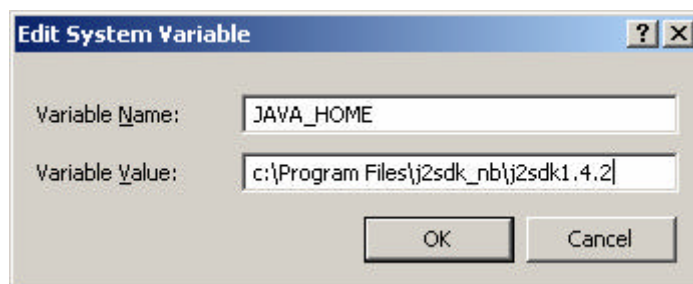
JDBC

### Goals:

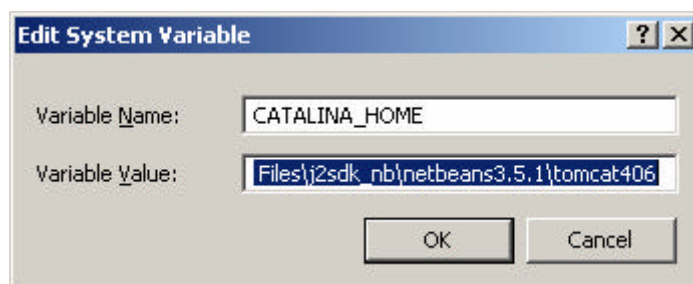
This tutorial explains how to get started using these important technologies. It takes you through running Tomcat standalone, creating your first JSP, creating a JSP to access a database and setting up NetBeans.

### Tutorial:

1. Set your JAVA\_HOME environment variable, if its is not already set. To do this go to the System applet in the control panel. Go to the advanced tab and click Environment Variables. It might be already set, but if not, Click New (System Variables) and enter:



2. Set your CATALINA\_HOME environment as above to be:



3. Before running Tomcat, we are going to create a new context to hold our new project. You should create a context for each project you want Tomcat to host.
4. Create a new folder called Student in the folder c:\Program Files\jdk\_nb\netbeans3.5.1\tomcat406\webapps. This is where we will be putting all our HTML and JSP files.
5. Now, use TextPad or Note Pad to edit the file: c:\Program Files\jdk\_nb\netbeans3.5.1\tomcat406\conf\Server.xml. This file holds all the configuration information for the Tomcat server.
6. Locate the line which reads:

```
<!-- Tomcat Root Context -->
```

```
<!--  
  <Context path="" docBase="ROOT" debug="0"/>  
-->
```

And add the following:

```
<Context path="/Student" docBase="Student" debug="0"/>
```

7. Now, in the Student folder which you previously created, create a new HTML file called index.html. Enter the following in the file:

```
<HTML>  
<BODY>  
  Hello world!  
</BODY>  
</HTML>
```

8. Open an MS DOS command prompt and navigate to the folder: c:\Program Files\jdk\_1.4.2\bin
9. Type startup and press enter. This will start the Tomcat server.
10. In Internet Explorer, type the URL: http://localhost:8080/Student/index.html (case sensitive).
11. All being well you should see the following page in IE.

Hello world!

Congratulations. You have successfully created your first Tomcat Project!

12. Now, in the same folder as index.html, create a file called index.jsp. Type the following into the file:

```
<HTML>  
<BODY>  
<%  
  out.println("Hello world from a JSP!");  
%>  
</BODY>  
</HTML>
```

13. In IE, navigate to http://localhost:8080/Student/index.jsp
14. All being well, you should see the following in IE:

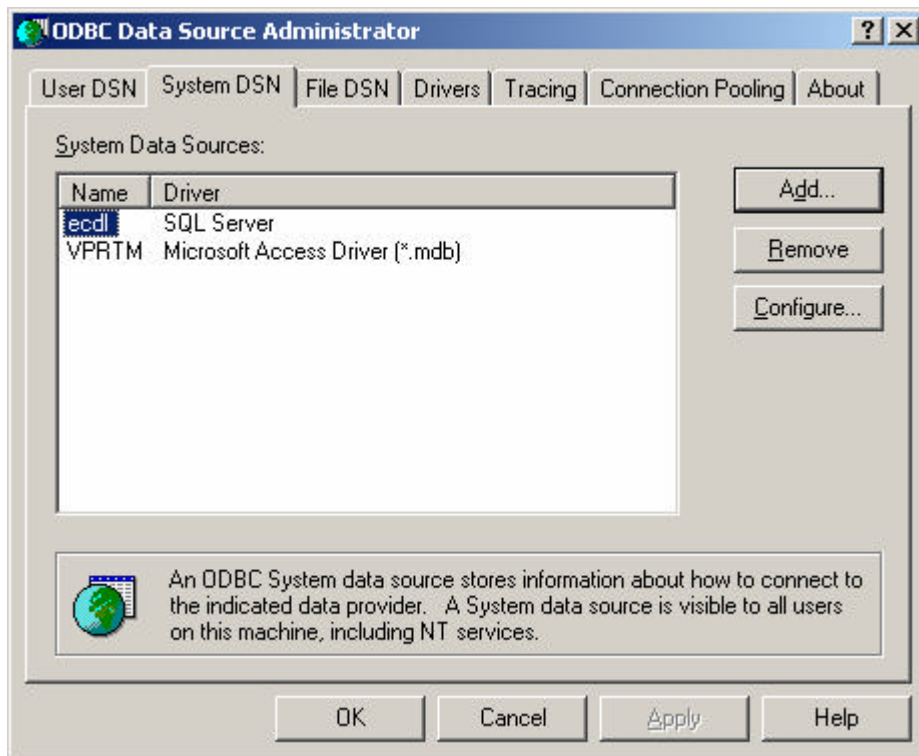
Hello world from a JSP!

You have now created your first JSP.

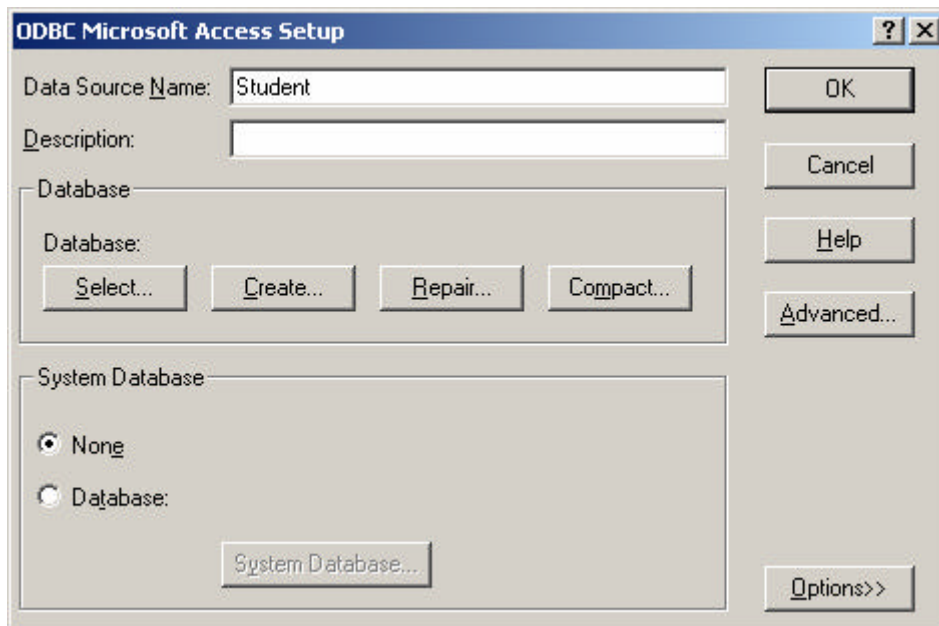
15. We will now move on to create a database and display the contents of a table in a web page. For this example, we will be using MS Access as the database, though the steps to use MySQL or Oracle or SQL Server would be similar.
16. Create a new Access database in the folder c:\Program Files\jdk\_1.4.2\bin\Student called Students.mdb.
17. Create a table in this database and create the following fields:

Table1 : Table	
Field Name	Data Type
StudentNumber	Text
Surname	Text
FirstName	Text
▶ DOB	Date/Time

18. Close the table design and call the table Student. When prompted to create a primary key, choose yes.
19. Now add some rows to the table. (10 will do).
20. Close Access.
21. We are going to use the ODBC/JDBC bridge driver to link up the JSP page to the database. Firstly we have to create an ODBC connection to the database. To do this, go to the Control Panel go to Administrative Tools and click Data Sources (ODBC).



22. Click System DSN and click Add.
23. Choose the Access driver and click finish. You will then see a screen like this one:



24. For database name, enter "Student".
25. Click Select and locate the database you previously created.
26. Now we need to create a JSP to list all the students.
27. Create a new JSP in the folder c:\Program Files\jdk\nb\netbeans3.5.1\tomcat406\webapps\Student called list\_students.jsp.
28. Use Notepad or TextPad to enter the following:

```
<%@ page import="java.sql.*" %>
```

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>
```

```
    A listing of all the students!
```

```
</TITLE>
```

```
<BODY>
```

```
<TABLE>
```

```
<%
```

```
    String url = "jdbc:odbc:Student";
```

```
    String uid = null;
```

```
    String pwd = null;
```

```
    Connection conn = null;
```

```
    ResultSet results = null;
```

```
    Statement statement = null;
```

```
    try {
```

```
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
        conn = DriverManager.getConnection(url, uid, pwd);
```

```
        statement = conn.createStatement();
```

```
        results = statement.executeQuery("select * from Student");
```

```
        while (results.next()) {
```

```
            out.print("<TR>");
```

```
            out.print("<TD>" + results.getInt("ID") + "</TD>");
```

```
            out.print("<TD>" + results.getString("Surname") + "</TD>");
```

```
            out.print("<TD>" + results.getString("FirstName") + "</TD>");
```

```

        out.println("<TD>" + results.getDate("DOB") + "</TD>");
    }
}
catch (ClassNotFoundException e) {
    System.out.println("Opps I can't find the JDBC Driver!");
    e.printStackTrace();
}
catch (SQLException e) {
    System.out.println("There was a problem with the SQL!");
    e.printStackTrace();
}
finally {
    if (conn != null) {
        try {
            conn.close();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}
%>
</TABLE>
<BODY>
<HTML>

```

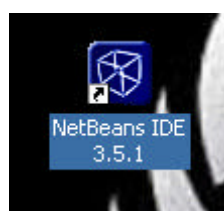
29. When you load the page: [http://localhost:8080/Student/list\\_students.jsp](http://localhost:8080/Student/list_students.jsp) in IE, you will see a table listing all the students in the database that you created.
30. In IE, Choose View | Source and you will notice that all the dynamic, server side code has been replaced by HTML:

```

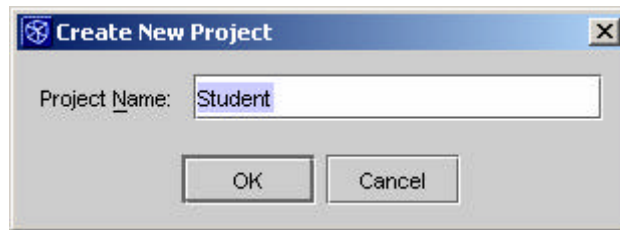
<BODY>
<TABLE>
<TR><TD>6</TD><TD>BERNEY</TD><TD>CONOR</TD><TD>1980-01-01</TD>
<TR><TD>7</TD><TD>BOLGER</TD><TD>JOHN</TD><TD>1980-01-01</TD>
<TR><TD>8</TD><TD>BROPHY</TD><TD>GARY</TD><TD>1980-01-01</TD>
<TR><TD>9</TD><TD>BURKE</TD><TD>GARETH</TD><TD>1980-01-01</TD>
<TR><TD>10</TD><TD>BURKE</TD><TD>KEVIN</TD><TD>1980-01-01</TD>
<TR><TD>11</TD><TD>CAHILL</TD><TD>LISA</TD><TD>1980-01-01</TD>
...

```

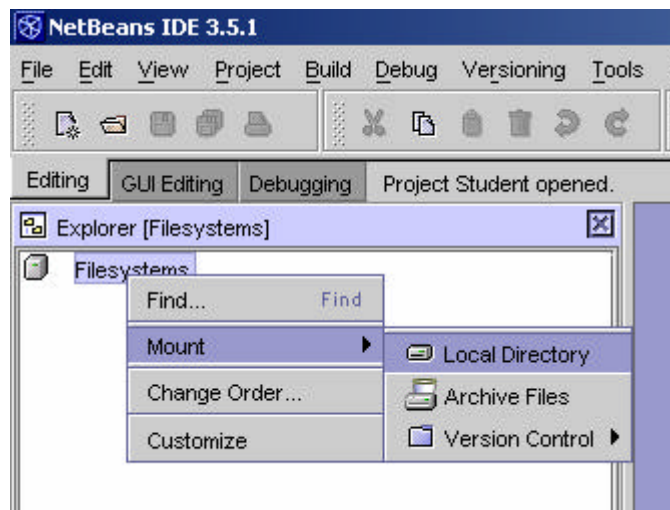
31. If you make a mistake typing in the code, Tomcat will throw an exception. You can then reedit the page and reload in your browser.
32. Now stop the Tomcat server by running shutdown.
33. Wouldn't it be great if there was an easier way of creating and maintaining JSP pages? Well there is... Enter NetBeans. Now lets create the same project using NetBeans.
34. Run the NetBeans IDE by clicking on the NetBeans icon:




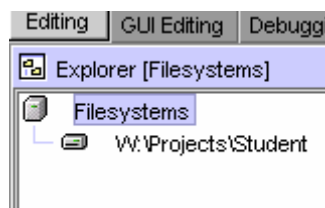
35. Firstly, we will create a new NetBeans project. To do this, choose Project | Project Manager | New and enter "Student" as the project name. Click OK.



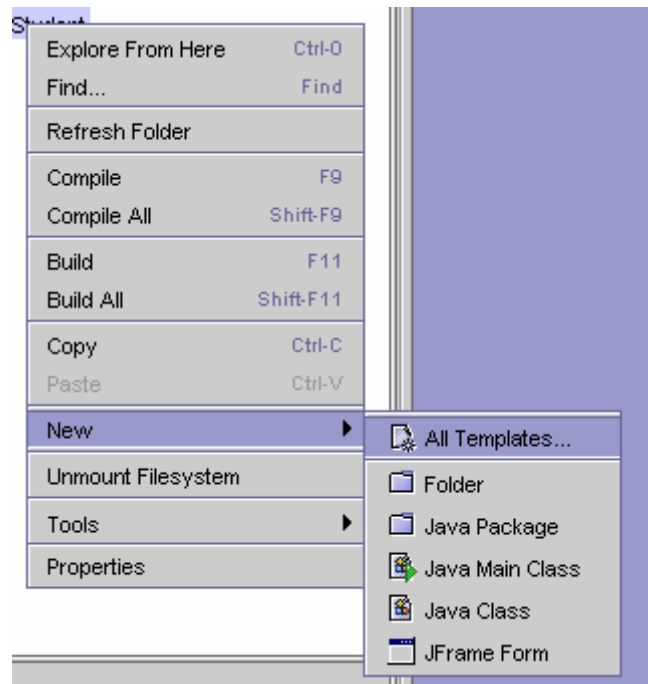
36. Now you have to mount a file system to give NetBeans a place to put your code. Right click on the Filesystems pane and choose Mount and then Local Directory.



37. Create a new folder to put your project into by clicking   
38. Name the folder "Student" and click Finish.  
39. Your Filesystems explorer pane should be updated with the new mounted file system:



40. Now right click on the folder, choose New and All Templates:



41. Select JSP and Servlets and choose JSP.
42. Name your JSP student\_list.jsp
43. Type in the code above, (or be clever and copy and paste it!). Some things to try:
44. NetBeans will help you write error free code, by using code completion. To activate code code completion on a line, press Ctrl + SPACE. This works for HTML, XML and JSPs!
45. NetBeans shows you the Javadocs for any classes in scope.
46. Once you have typed in your code, you can test your JSP by simply right clicking on the file and clicking Execute. NetBeans will start an Internal Tomcat instance and load the web page in the default Browser.
47. NetBeans may prompt you to convert your project to a web project. If it does, just choose yes.