Practical 06 - Build and deploy a WebService application

Objectives:

- To build a functioning WebService in Eclipse.
- To publish the WebService as an application on TomCat
- To view the WebService WSDL
- The web service should provide the following functions:
- The web service should provide the following functions:
    - If given the radius of a circle, returns the area of the circle (pi X radius X radius)
    - If given the length and breadth of rectangle, returns the area of the rectangle (length x breadth)

Requirements for today:

- Oracle Java JDK v1.7 (aka Java 7) - www.java.com (already installed in the VM provided)
- Eclipse JEE Kepler - www.eclipse.org
- Apache Tomcat - www.apache.org
- SOAPUI (free version, not Pro)- www.soapui.org
- ...all installed as per Practical 01.

Instructions:

1. Launch Virtual Machine, launch Eclipse
    1.1. Start and sign on to the Virtual Machine for this course.
    1.2. Locate the "eclipse" folder where you installed the Eclipse JEE IDE (typically /home/user/development/eclipse)
    1.3. Within that folder, locate file "eclipse", and double-click it.
    1.4. In the "Execute File" dialogue, click on <Execute>
    1.5. If you are asked about the location for the Workspace, choose a suitable directory (the default is fine), then click <OK>.
    1.6. Wait for Eclipse to finish launching.
    1.7. If the "Welcome" page appears, close it.
2. Unpack the TomCat server
    2.1. Launch the file manager from the “Start” bar.
    2.2. Locate the Tomcat installation file (generally in /home/user/Downloads/apache-tomcat-X.X.X.tar-gz or similar)
    2.3. Double click the Tomcat installation file.

2.4. When the Archive Manager opens, extract the folder “apache-tomcat-x.x.x” to the folder /home/development.
2.5. Close the Archive Manager.

3. Tell Eclipse about the Tomcat server
   3.1. In the Eclipse window, select "Window" then "Preferences"
   3.2. In the "Preferences" dialog, open the "Server" item, then select "Runtime Environments"
   3.3. In the "Server Runtime Environments" dialog, click <Add>
   3.4. In the "New Server Runtime Environment" dialog:
      3.4.1. Open the "Apache" folder
      3.4.2. Select "Apache Tomcat v7.0" item.
      3.4.3. Check that "Create a new local server" is not selected.
      3.4.4. Click <Next>
   3.5. In the "Tomcat Server" dialog:
      3.5.1. Leave "Name" and "JRE" untouched.
      3.5.2. For the filed "Tomcat installation directory", click Browse:
      3.5.3. In the dialog that appears, select directory you extracted Tomcat into above (typically /home/user/development/apache-tomcat-x-x-x or similar).
      3.5.4. Click <OK>
   3.6. Back at the "Tomcat Server" dialog, click "Finish"
   3.7. Back at the "Preferences" dialog, click "OK"
4. Create a new project to hold our application
   4.1. Back at the Eclipse desktop right click in a blank area of the "Project Explorer" window.
   4.2. Select "New", then "Project"
   4.3. In the "New Project" dialog:
      4.3.1. Locate and open the "Web" folder.
      4.3.2. In the "Web" folder, locate and select the "Dynamic Web Project" item.
      4.3.3. Click <Next>
   4.4. In the "New Dynamic Web Project" dialog:
      4.4.1. Enter the project name "Practical06" – Note: no spaces in name.
      4.4.2. Note that the "Target Runtime" is set to the "Apache Tomcat v7.0" instance we created earlier
      4.4.3. Click <Next>
   4.5. At the "Java" dialog, click <Next>
   4.6. At the "Web Module" dialog:
      4.6.1. Write down the value in "Context Root" here: - NB: Case Sensitive! :_____
      4.6.2. Ensure that "Generate web.xml deployment descriptor" is selected.
      4.6.3. Click <Finish>
   4.7. Wait for the Dynamic Web Project to be created

4.8. Back in the "Project Explorer" window:
  4.8.1. Locate the dynamic web project we just created, and open it.
  4.8.2. Note the folders and other items created to support the project.

5. Create a new Java Class containing our code
  5.1. Back at the Eclipse desktop, locate the new Dynamic Web Project in the "Project Explorer" window.
  5.2. Open the Dynamic Web Project
  5.3. Within the Dynamic Web Project, locate and open the "Java Resources" folder.
  5.4. Within the "Java Resources" folder, locate the "src" folder.
    5.4.1. Right click the "src" folder, then select "New" then "Class"
  5.5. In the "New Java Class" dialog:
    5.5.1. Set the "Package" name ("ie.ipa.webservices.<username>" is appropriate, replaing <username> above with your username, e.g. jsmith
    5.5.2. Set the "Class" name (e.g. "ShapeAreas") write it down here: - NB: Case Sensitive! : _____
    5.5.3. Leave all the other defaults unchanged.
    5.5.4. Click <Finish>
  5.6. Back at "src" folder, note that the package and .java file have been created, and the .java file has been opened for editing:

6. Add code to the .java file:
  6.1. The .java file currently contains:

```
package <package name, as set>
public class <class name as set> {
}
```

  6.2. Add code so it looks like the following:

```
package <package name, as set>
public class <class name as set> {
public double circleArea( double radius ){
return radius * radius * 3.14 ;
}
public double rectangleArea( double length , double breadth ){
return length * breadth ;
}
public String helloName(String name){
return "Hello " + name;
}
}
```

7. Check and review the code:
  7.1. Save and validate the code you types in, fix any errors. Ensure that the validation completes with no errors or warnings before proceeding

7.2. Review the code, and try and figure out what it does before proceeding...
7.3. Close the editor window for "ShapeAreas.java"

8. Create a new web service based on this code.
   8.1. In the Package Explorer window, locate the "ShapeAreas.java" file.
   8.2. Right click the ".java" file, then select "New" then "Other"
   8.3. In the "New" dialog:
      8.3.1. Locate and open the "Web Services" folder
      8.3.2. Within the "Web Services" folder, locate then select "Web Service" item.
      8.3.3. Click <Next>
   8.4. Within the "Web Service" dialog:
      8.4.1. The top section refers to the WebService we are creating:
      8.4.2. Note that the "Web service type" is set to "Bottom up Java bean Web Service"
      8.4.3. Check that "Service Implementation" is set to the package name and class name you set earlier.
      8.4.4. Check that the top "slider" is set to the value "Start Service"
      8.4.5. The lower section refers to a "client" we can create for this WebService – we don't want this at the moment, so check that the bottom "slider" is set to the lowest value, i.e. "Configuration: No Client Generation"
      8.4.6. Ensure that "Publish the Web Service" is selected.
      8.4.7. Click <Next>
   8.5. Within the "Web Service Java Bean Identity" dialog.
      8.5.1. Ensure that all the methods in your class are selected methods.
      8.5.2. Note the name of the WSDL file.
      8.5.3. Leave all other settings at default.
      8.5.4. Click <Next>
   8.6. At the "Server Startup" dialog:
      8.6.1. Click <Start server>
      8.6.2. Wait for the server (the Tomcat instance we specified) to start.
      8.6.3. Click <Next>
      8.6.4. Wait for the deployment to complete.
   8.7. At the "Web Service Publication":
      8.7.1. De-select both options (we're not using UDDI today)
      8.7.2. Click "Finish"
   8.8. Leave Eclipse running - don't close the application!
9. Use your browser to examine the WebService and WSDL
   9.1. Launch a web browser.
   9.2. Enter the address of http://localhost:8080/<context root>/services/<class name>
   9.3. Note that there is a dummy "holding" page there.

9.4. Enter the address of http://localhost:8080/<context root>/services/<class name>?WSDL
9.5. Note the returned WSDL is based on the code in the .class file
9.6. In particular, note the type, messages, operations and ports - make sure you can see how these relate to the code.
9.7. Highlight and copy this URL - also, write it down here: - NB: Case Sensitive!
9.8. WEB_SERVICE_URL = _____

10. Export the complete application as a WAR file
    10.1. Return to the Eclipse Project Explorer window
    10.2. Locate the main "Dynamic Web Project" we have created.
    10.3. Right click the Project
    10.4. Select "Export" then "WAR file"
    10.5. In the "WAR Export" dialogue
    10.6. Select a destination file for the WAR export
    10.7. Note the available options
    10.8. Click Finish
11. Investigate the .war file:
    11.1. Locate the exported ".war" file.
    11.2. Double click the ".war" file to open this file with the Archive Manager and examine the contents.