Create a Servlet

There are three ways to create the servlet.

1. By implementing the Servlet interface
2. By inheriting the GenericServlet class
3. By inheriting the HttpServlet class

The HttpServlet class is widely used to create the servlet because it provides methods to handle http requests such as doGet(), doPost, doHead() etc.

In this example we are going to create a servlet that extends the HttpServlet class. In this example, we are inheriting the HttpServlet class and providing the implementation of the doGet() method. Notice that get request is the default request.

sai@node2:~/apache-tomcat-7.0.84$ cd webapps/

sai@node2:~/apache-tomcat-7.0.84/webapps$ mkdir Demo

sai@node2:~/apache-tomcat-7.0.84/webapps$ cd Demo

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ ls

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ mkdir src

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ mkdir WEB-INF

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ ls

src WEB-INF

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ cd WEB-INF

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/WEB-INF$ mkdir classes

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/WEB-INF$ ls

classes

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/WEB-INF$ cd ..

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ ls

src WEB-INF

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ cd src

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/src$ gedit DemoServlet.java

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/src$ ls

DemoServlet.java DemoServlet.java~

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/src$ javac -d ../WEB-INF/classes/ DemoServlet.java

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/src$ cd ..

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo$ cd WEB-INF/

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/WEB-INF$ gedit web.xml

sai@node2:~/apache-tomcat-7.0.84/webapps/Demo/WEB-INF$

DemoServlet.java

```java
1. import javax.servlet.http.*;
2. import javax.servlet.*;
3. import java.io.*;
4. public class DemoServlet extends HttpServlet{
5. public void doGet(HttpServletRequest req,HttpServletResponse res)
   throws ServletException,IOException
6. {
7. res.setContentType("text/html");/* setting the content type This will be returned to the browser */
8. PrintWriter pw=res.getWriter();//get the stream to write the data
9. pw.println("<html><body>");  //writing html in the stream
10.pw.println("Welcome to servlet");
11.pw.println("</body></html>");
12.
13.pw.close();//closing the stream
14.}}
```

Description of the elements of web.xml file

There are too many elements in the web.xml file. Here is the illustration of some elements that is used in the above web.xml file. The elements are as follows:

<web-app> represents the whole application.

<servlet> is sub element of <web-app> and represents the servlet.

<servlet-name> is sub element of <servlet> represents the name of the servlet.

<servlet-class> is sub element of <servlet> represents the class of the servlet.

<servlet-mapping> is sub element of <web-app>. It is used to map the servlet.

<url-pattern> is sub element of <servlet-mapping>. This pattern is used at client side to invoke

the servlet.

Extending Generic Class

sai@node2:~$ cd apache-tomcat-7.0.84/

sai@node2:~/apache-tomcat-7.0.84$ cd webapps/

sai@node2:~/apache-tomcat-7.0.84/webapps$ mkdir EGC

sai@node2:~/apache-tomcat-7.0.84/webapps$ cd EGC/

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC$ mkdir src

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC$ mkdir -p WEB-INF/classes

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC$ ls

src WEB-INF

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC$ cd src

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/src$ gedit First.java

```
import java.io.*;
import javax.servlet.*;

public class First extends GenericServlet{
public void service(ServletRequest req,ServletResponse res)
throws IOException,ServletException{

res.setContentType("text/html");

PrintWriter out=res.getWriter();
out.print("<html><body>");
out.print("<b>hello generic servlet</b>");
out.print("</body></html>");

}
```

}

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/src$ javac -d ../WEB-INF/classes/ First.java

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/src$ ls

First.java

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/src$ cd ..

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC$ ls

src WEB-INF

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC$ cd WEB-INF/classes/

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/WEB-INF/classes$ ls

First.class

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/WEB-INF/classes$ cd ..

sai@node2:~/apache-tomcat-7.0.84/webapps/EGC/WEB-INF$ gedit web.xml

<web-app>
 <servlet>
  <servlet-name>GenericExample</servlet-name>
  <servlet-class>First</servlet-class>
 </servlet>
 <servlet-mapping>
  <servlet-name>GenericExample</servlet-name>
  <url-pattern>/ge</url-pattern>
 </servlet-mapping>
</web-app>