(varianta pdf)

SwRTc – supliment proiect

Tehnologia *Java Server Pages* (JSP)

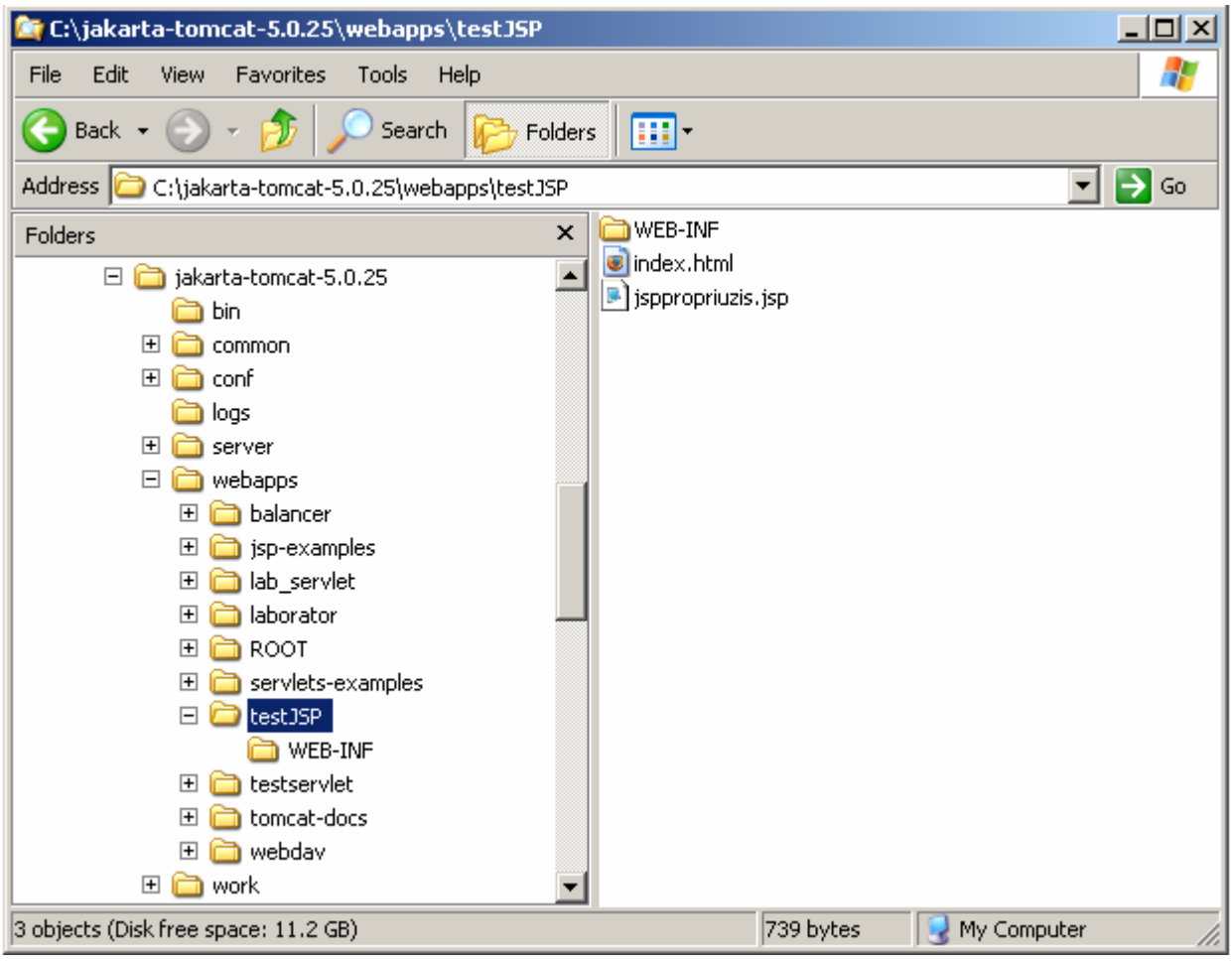
JSP.1. Descrierea

In aceasta lucrare vor fi acoperite (partial) urmatoarele probleme:
- instalarea, compilarea si invocarea JSP-urilor Java
- exemple de *servlet-uri* si JSP-uri

JSP.2. Exemplu simplu de JSP

JSP.2.1. Structura de directoare si continuturile fisierelor (structura instalarii JSP-ului)

Structura de directoare pentru un JSP simplu (plasat in subdirectorul testJSP al directorului webapps)



Directorul WEB-INF este necesar, dar el poate sa nu contina nimic. In cazul in care directorul WEB-INF nu contine un fisier web.xml, un astfel de fisier este generat automat de TOMCAT.

Mesajele afisate la startup in acest caz sunt:

```
Jan 8, 2006 3:09:32 PM org.apache.catalina.core.StandardHostDeployer install
INFO: Installing web application at context path /testJSP from URL file:C:\jakarta-
tomcat-5.0.25\webapps\testJSP
Jan 8, 2006 3:09:32 PM org.apache.catalina.startup.ContextConfig applicationConfig
INFO: Missing application web.xml, using defaults only
StandardEngine[Catalina].StandardHost[localhost].StandardContext[/testJSP]
```

Continutul fisierului index.html

```
 1 <html>
 2   <head>   <title>Index pentru testJSP</title>  </head>
 3   <body>
 4     <p><h1>Index pentru testJSP</h1>
 5     <form action = jsppropriuzis.jsp method = get>
 6         <label><p>Your name:
 7         <input type="text" size="20" name="name" value="anonymous">
 8         <p><input type = "submit" value = "Say Hello"/>
 9         </label></p>
10     </form>
11     <p>Acest fisier poate fi accesat la adresa:
12        <a href="http://localhost:8080/testJSP/">http://localhost:8080/testJSP/</a>
13   </body>
14 </html>
```

Continutul fisierului jsppropriuzis.jsp

```
 1 <html>
 2   <head>   <title>JSP simplu</title>  </head>
 3   <body bgcolor="white">
 4     <%-- Acesta este un comentariu JSP --%>
 5     <h1> Hello <%=request.getParameter("name") %>! </h1>
 6   </body>
 7 </html>
```

In momentul interpretarii JSP-ului jsppropriuzis.jsp containerul servlet/JSP al TOMCAT:

- mai intai va genera automat un servlet, numit jsppropriuzis_jsp.java,

- apoi va compila automat acest servlet, obtinandu-se fisierul clasa al servlet-ului, numit jsppropriuzis_jsp.class,

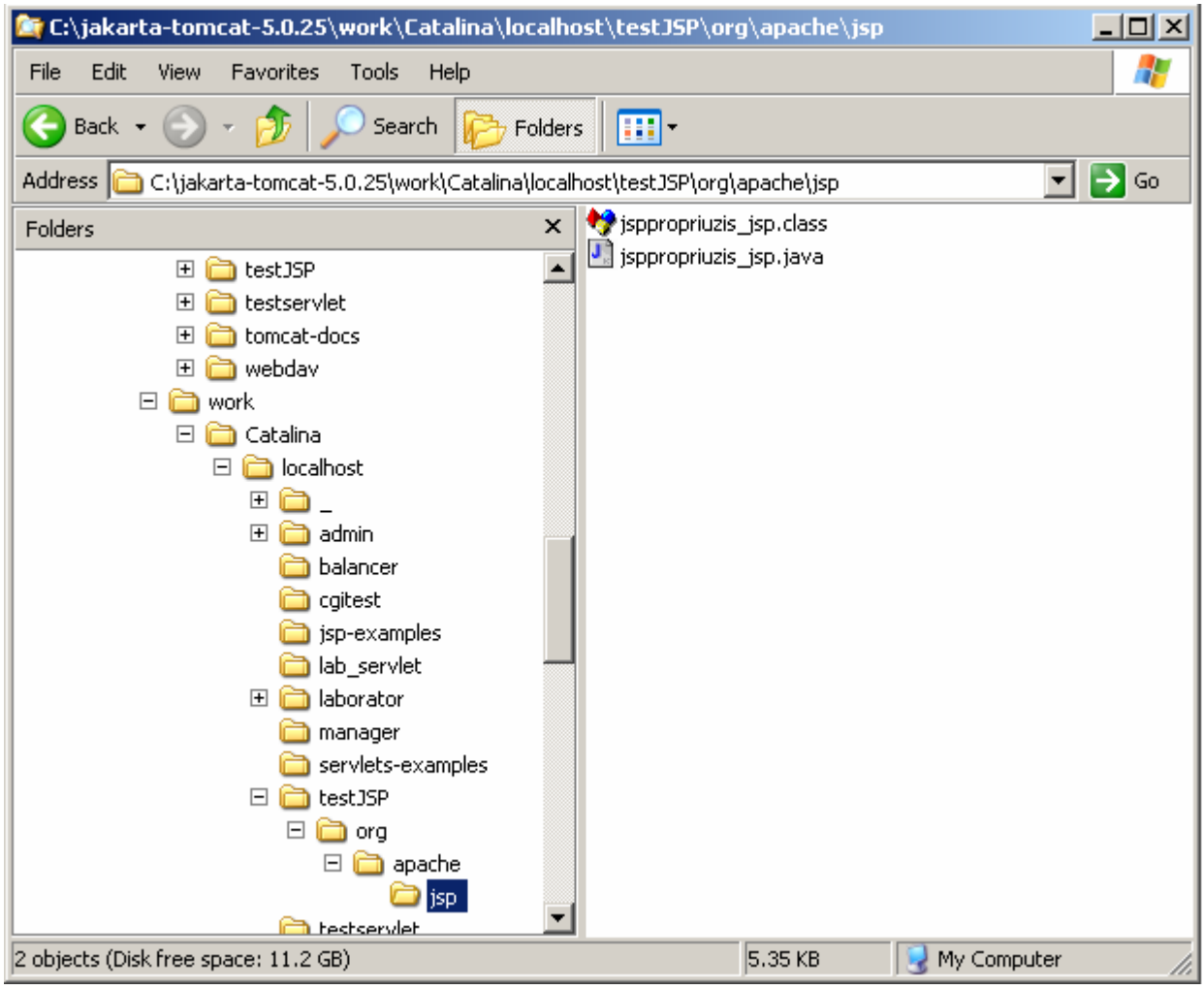
- in final va executa acest servlet.

Continutul fisierului jsppropriuzis_jsp.java

```
 1 package org.apache.jsp;
 2 import javax.servlet.*;
 3 import javax.servlet.http.*;
 4 import javax.servlet.jsp.*;
 5
 6 public final class jsppropriuzis_jsp extends org.apache.jasper.runtime.HttpJspBase
 7                     implements org.apache.jasper.runtime.JspSourceDependent {
 8   private static java.util.Vector _jspx_dependants;
 9
10   public java.util.List getDependants() {
11     return _jspx_dependants;
12   }
13   public void _jspService(HttpServletRequest request, HttpServletResponse response)
```

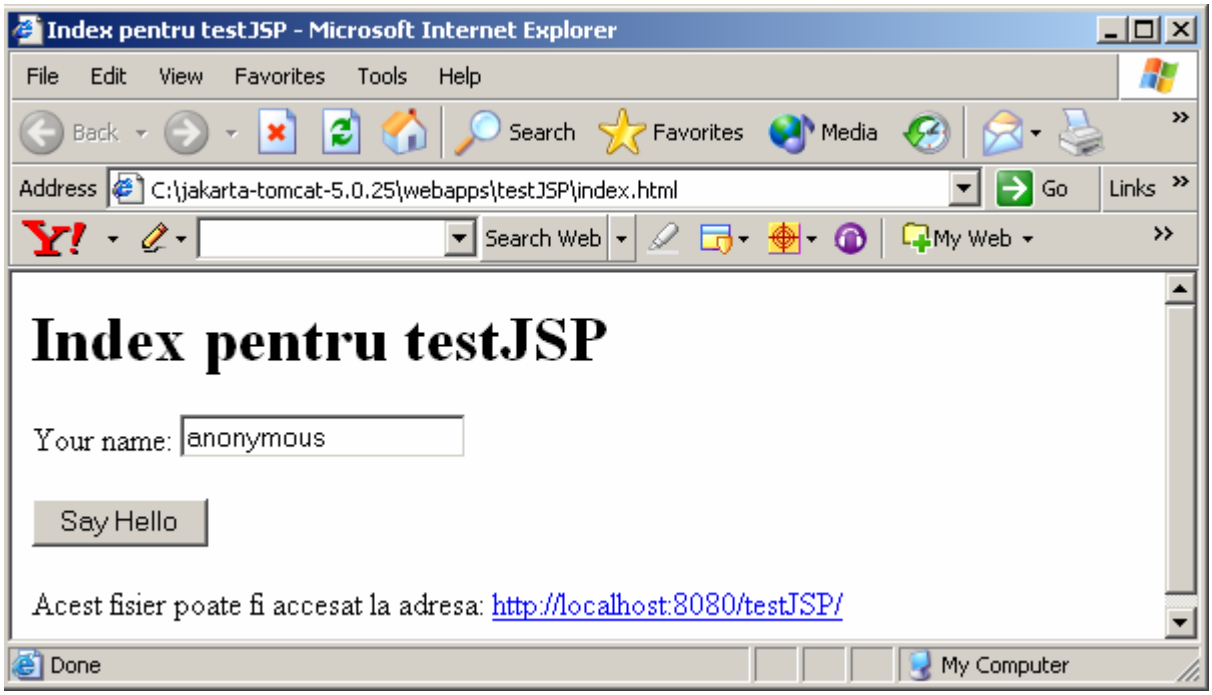
```
14          throws java.io.IOException, ServletException {
15      JspFactory _jspxFactory = null;
16      PageContext pageContext = null;
17      HttpSession session = null;
18      ServletContext application = null;
19      ServletConfig config = null;
20      JspWriter out = null;
21      Object page = this;
22      JspWriter _jspx_out = null;
23      PageContext _jspx_page_context = null;
24
25      try {
26        _jspxFactory = JspFactory.getDefaultFactory();
27        response.setContentType("text/html");
28        pageContext = _jspxFactory.getPageContext(this, request, response,
29                      null, true, 8192, true);
30        _jspx_page_context = pageContext;
31        application = pageContext.getServletContext();
32        config = pageContext.getServletConfig();
33        session = pageContext.getSession();
34        out = pageContext.getOut();
35        _jspx_out = out;
36
37        out.write("<html>\r\n");
38        out.write("  <head>\r\n");
39        out.write("   <title>JSP simplu</title>\r\n");
40        out.write("  </head>\r\n");
41        out.write("  <body bgcolor=\"white\">\r\n");
42        out.write("    ");
43        out.write("\r\n");
44        out.write("\r\n");
45        out.write("    <h1> Hello ");
46        out.print(request.getParameter("name") );
47        out.write("! </h1>\r\n");
48        out.write("\r\n");
49        out.write("  </body>\r\n");
50        out.write("</html>\r\n");
51      } catch (Throwable t) {
52        if (!(t instanceof SkipPageException)){
53          out = _jspx_out;
54          if (out != null && out.getBufferSize() != 0)
55            out.clearBuffer();
56          if (_jspx_page_context != null) _jspx_page_context.handlePageException(t);
57        }
58      } finally {
59        if (_jspxFactory !=null) _jspxFactory.releasePageContext(_jspx_page_context);
60      }
61    }
62  }
```

Structura de directoare pentru servlet-ul generat de containerul JSP in momentul interpretarii JSP-ului jsppropriuzis.jsp

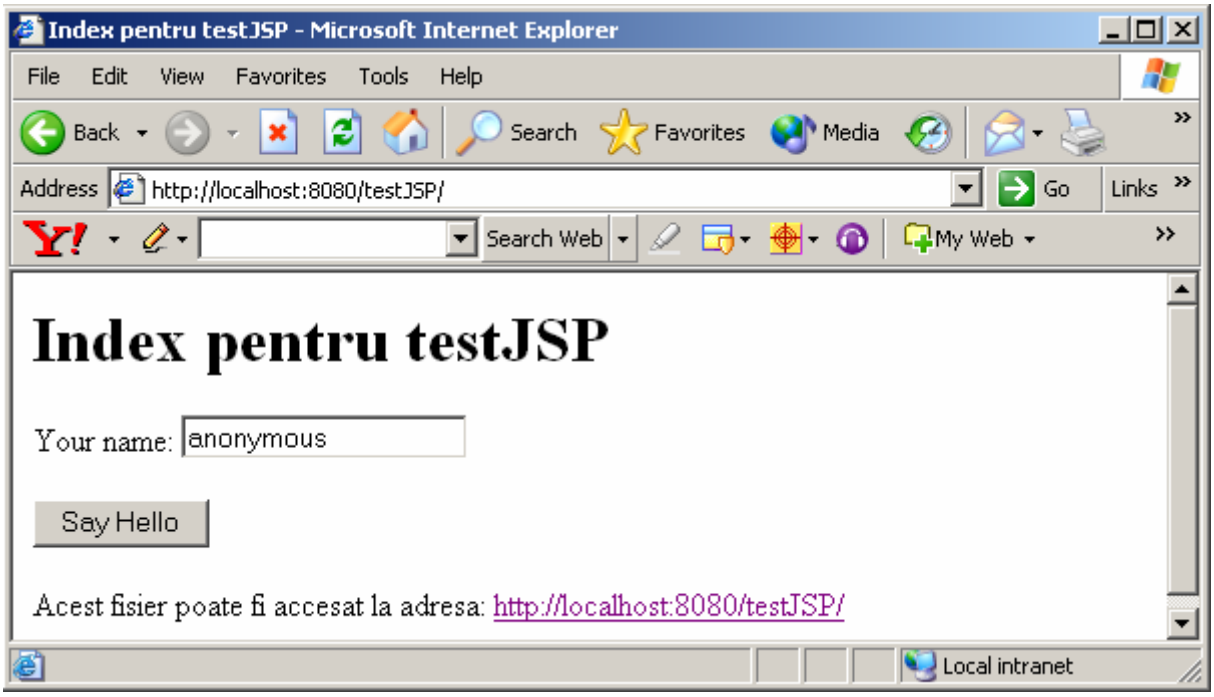


JSP.2.2. Utilizarea JSP-ului

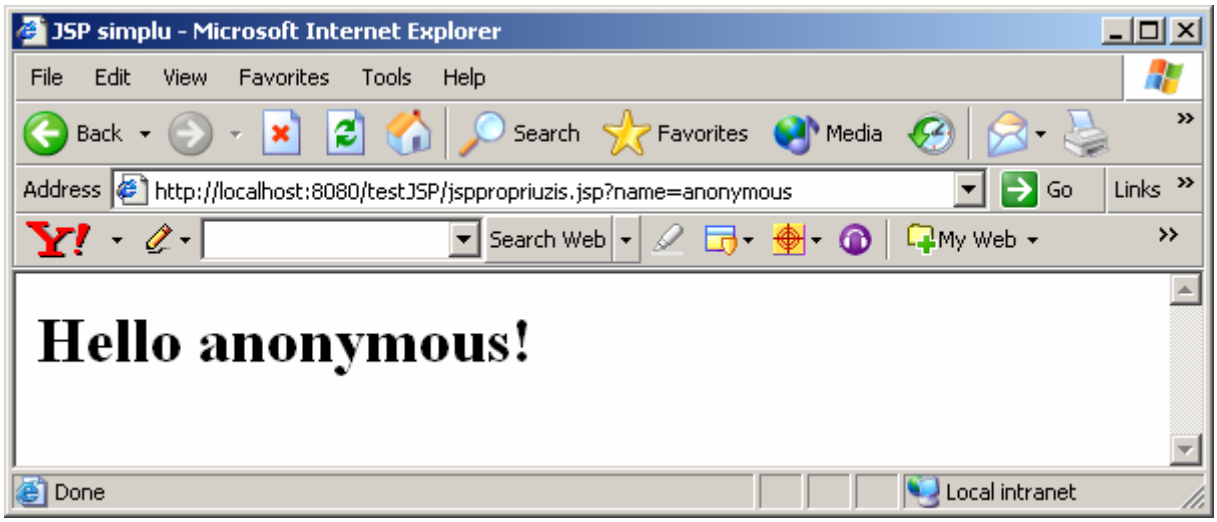
Fisierul index.html incarcat local in browser



Fisierul index.html incarcat in browser de pe server (utilizand URL-ul `http://localhost:8080/testJSP/`)



Dupa apelul GET catre JSP (obtinut prin apasarea butonului “Say Hello”)



Continutul html al browserului (obtinut cu *View Source*) pentru pagina sursa jsppropriuzis.jsp

```
1  <html>
2    <head>
3     <title>JSP simplu</title>
4    </head>
5    <body bgcolor="white">
6      <h1> Hello anonymous! </h1>
7    </body>
8  </html>
```

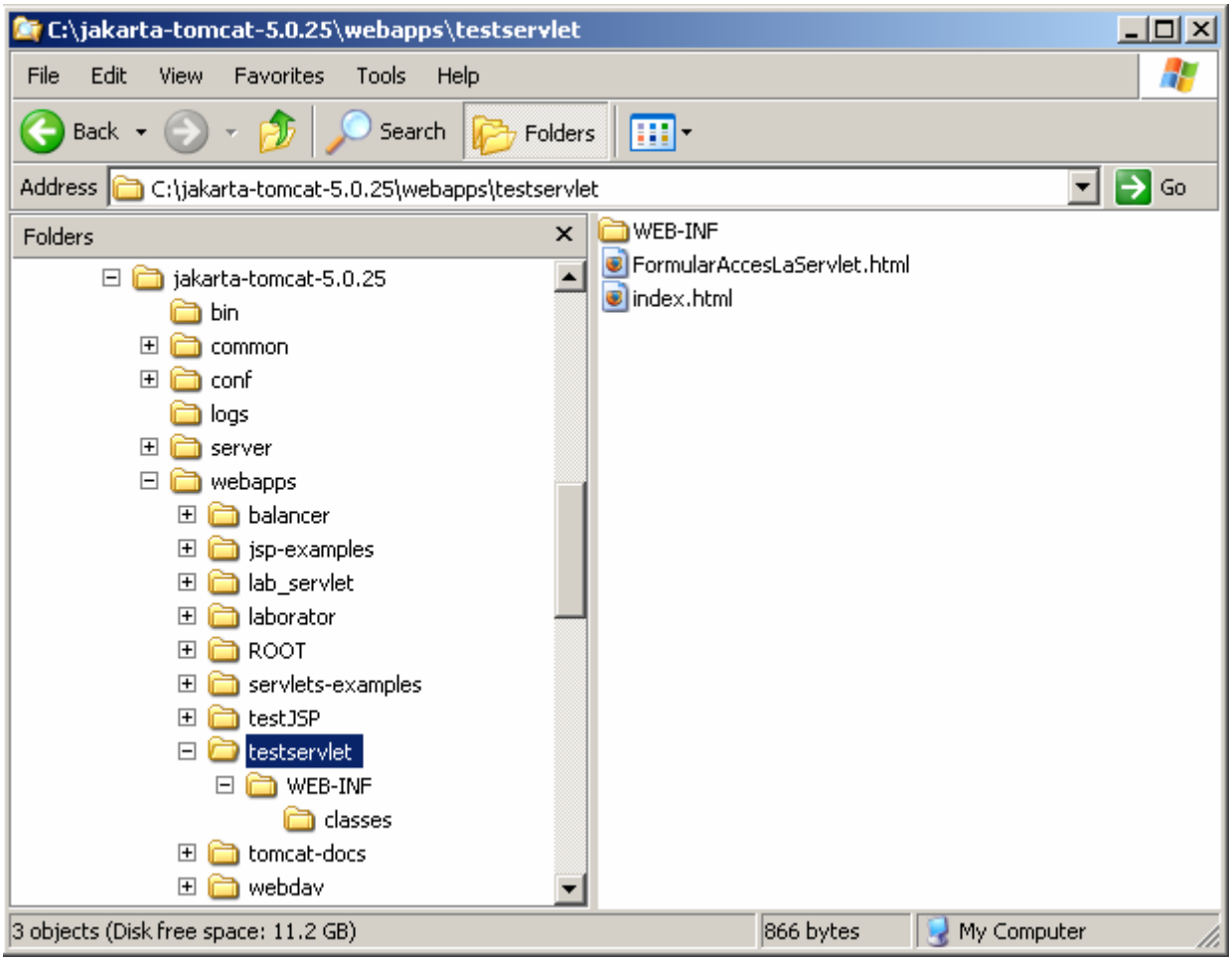
Distributia testJSP.zip (arhiva care contine instalarea JSP-ului).

JSP.3. Exemplu simplu de servlet

Pentru o comparatie intre JSP-uri si *servlet-uri*, urmatorul exemplu simplu de *servlet* realizeaza aceeasi functionalitate ca exemplul simplu de JSP anterior.

JSP.3.1. Structura de directoare si continuturile fisierelor (structura instalarii *servlet-ului*)

Structura de directoare pentru un servlet simplu (plasat in subdirectorul testservlet al directorului webapps)



Continutul fisierului index.html

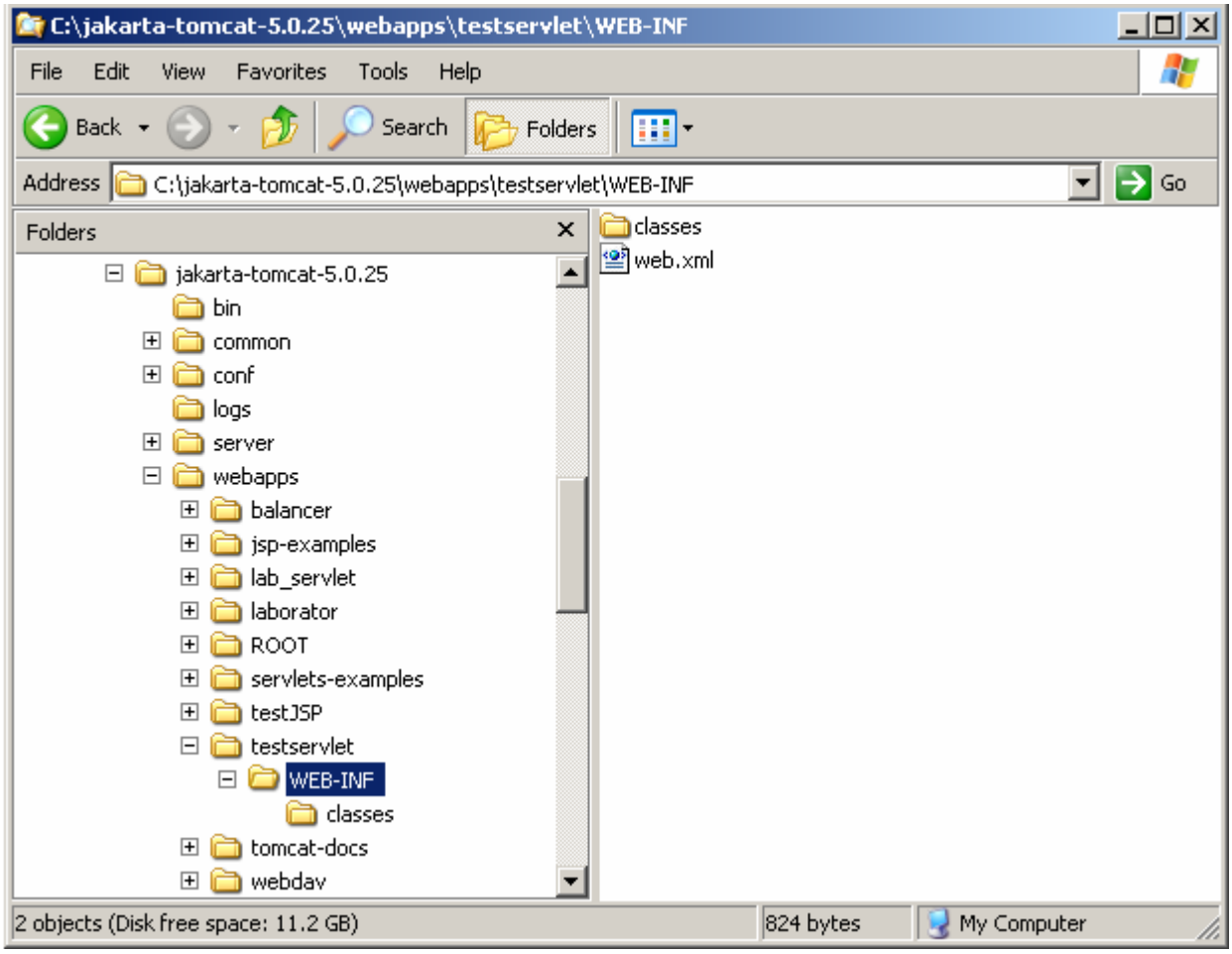
```
 1  <html>
 2    <head>   <title>Index pentru testservlet</title>  </head>
 3    <body>
 4      <p><h1>Index pentru testservlet</h1>
 5         <a href="http://localhost:8080/testservlet/FormularAccesLaServlet.html">
 6                <b>Formularul de acces la servlet</b></a>
 7
 8         <p>Acest fisier poate fi accesat la adresa:
 9              <a href="http://localhost:8080/testservlet/">
10               http://localhost:8080/testservlet/</a>
11    </body>
12  </html>
```

Continutul fisierului FormularAccesLaServlet.html

```
 1  <html>
 2    <head>   <title>Formular de acces la servlet</title>  </head>
```

```
 3   <body>
 4     <p><h1>Formular de acces la servlet</h1>
 5     <form action = "servlet/numeleservletului" method = "get"
 6         <label><p>Your name:
 7         <input type="text" size="20" name="name" value="anonymous">
 8         <p><input type = "submit" value = "Say Hello"/>
 9         </label></p>
10     </form>
11   </body>
12 </html>
```

Continutul subdirectorului WEB-INF

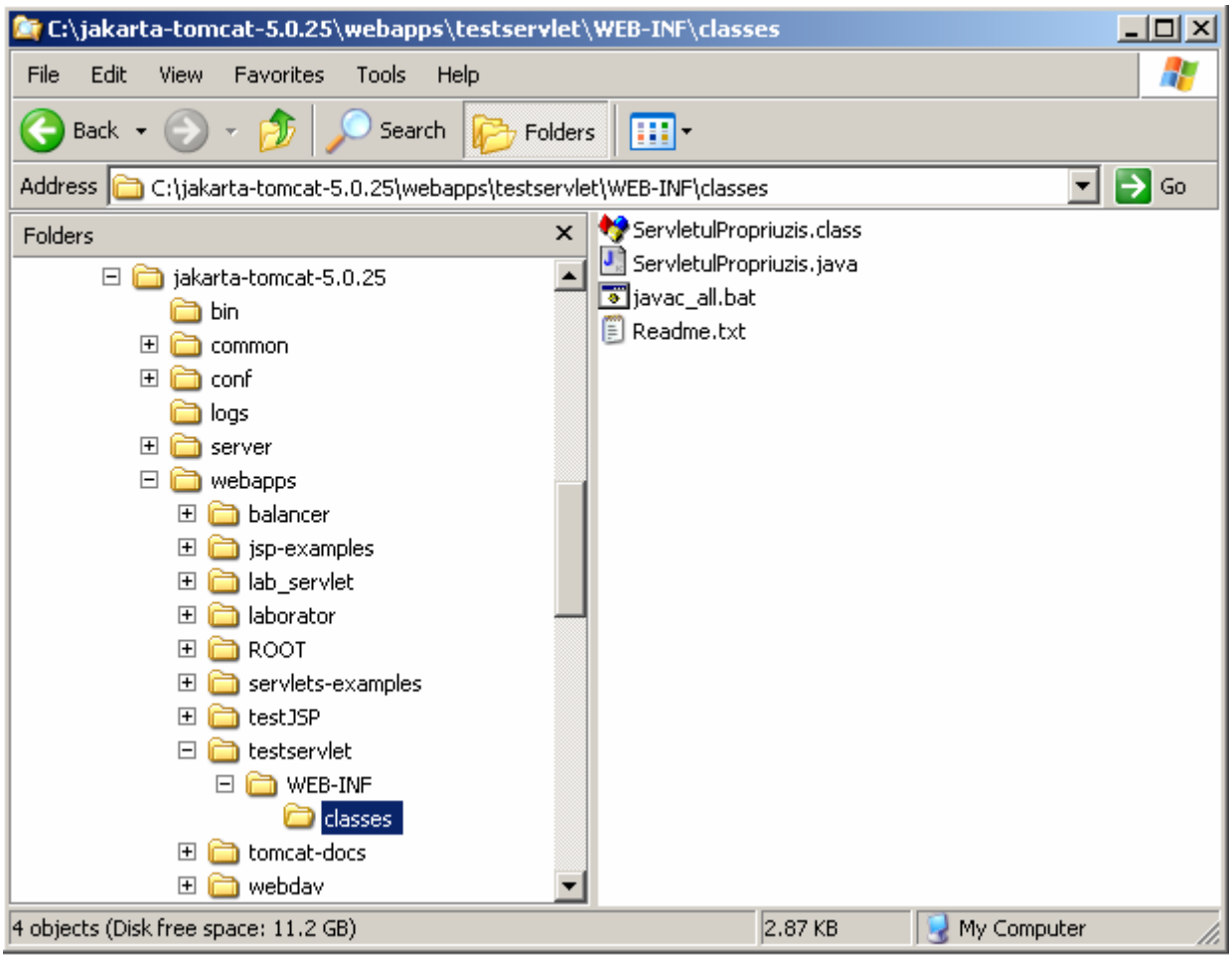


Continutul fisierului web.xml

```
 1 <?xml version="1.0" encoding="ISO-8859-1"?>
 2
 3 <!DOCTYPE web-app
 4     PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 5     "http://java.sun.com/dtd/web-app_2_3.dtd">
 6
 7 <web-app>
 8
 9   <!--  Descriere generala a aplicatiei Web  -->
10     <display-name>Exemplu simplu</display-name>
11     <description>
12         Exemplu simplu
13     </description>
14
15   <!--  Definitii (asocierea numelor cu clasele Java) -->
16     <servlet>
17         <servlet-name>numeleservletului</servlet-name>
18         <servlet-class>ServletulPropriuzis</servlet-class>
19     </servlet>
20
21   <!--  Translatii (asocierea numelor cu schemele de invocare) -->
```

```
22      <servlet-mapping>
23          <servlet-name>numeleservletului</servlet-name>
24          <url-pattern>/servlet/numeleservletului</url-pattern>
25      </servlet-mapping>
26
27  </web-app>
```

Continutul subdirectorului classes



Continutul fisierului ServletulPropriuzis.java

```
 1  import java.io.*;
 2  import java.text.*;
 3  import java.util.*;
 4  import javax.servlet.*;
 5  import javax.servlet.http.*;
 6
 7  public class ServletulPropriuzis extends HttpServlet {
 8
 9      public void doGet(HttpServletRequest cerereHTTP,
10              HttpServletResponse raspunsHTTP) throws IOException, ServletException {
11
12          String name = (String) cerereHTTP.getParameter("name");
13
14          raspunsHTTP.setContentType("text/html");
15
16          PrintWriter outHTTP = raspunsHTTP.getWriter();
17
18          outHTTP.println("<html>");
19          outHTTP.println("<head>");
20          outHTTP.println("<title> Hello " + name + "! </title>");
21          outHTTP.println("</head>");
22          outHTTP.println("<body bgcolor=\"white\">");
23          outHTTP.println("<h1> Hello " + name + "! </h1>");
24          outHTTP.println("</body>");
25          outHTTP.println("</html>");
```

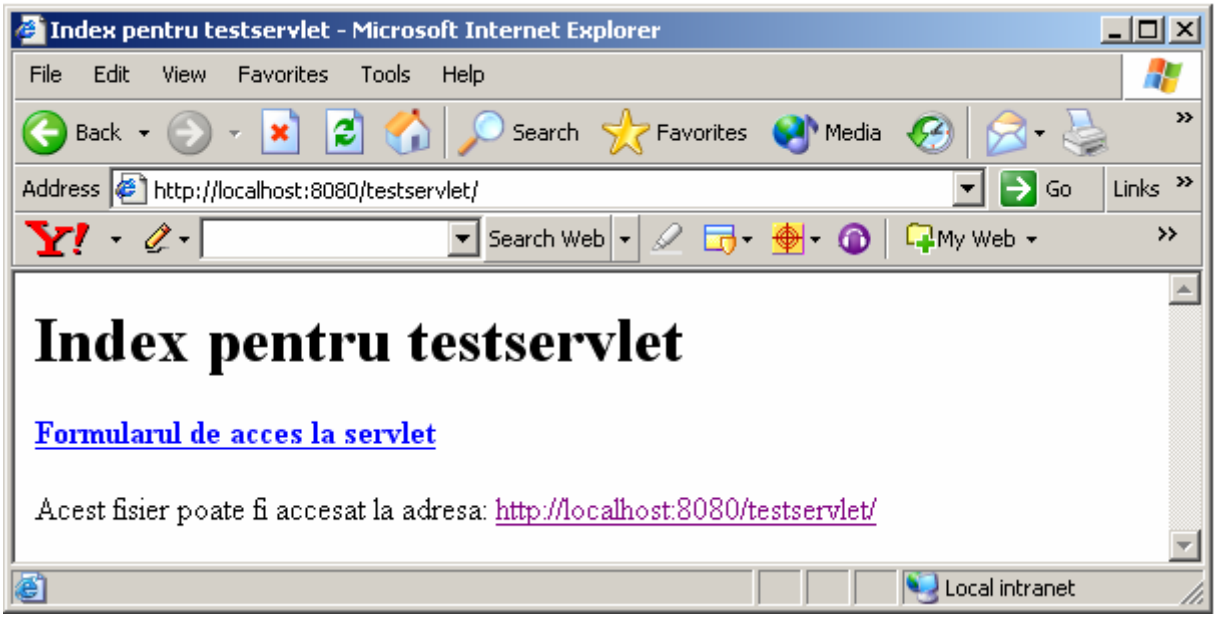
```
26      }
27  }
```

Fisierul sursa al servlet-ului, numit ServletulPropriuzis.java, trebuie compilat (de exemplu utilizand script-ul javac_all.bat, prin executia caruia sunt compilate toate sursele Java din directorul curent) obtinandu-se fisierul clasa al servlet-ului, numit ServletulPropriuzis.class.

In momentul cererii GET servlet-ul ServletulPropriuzis va fi executat.

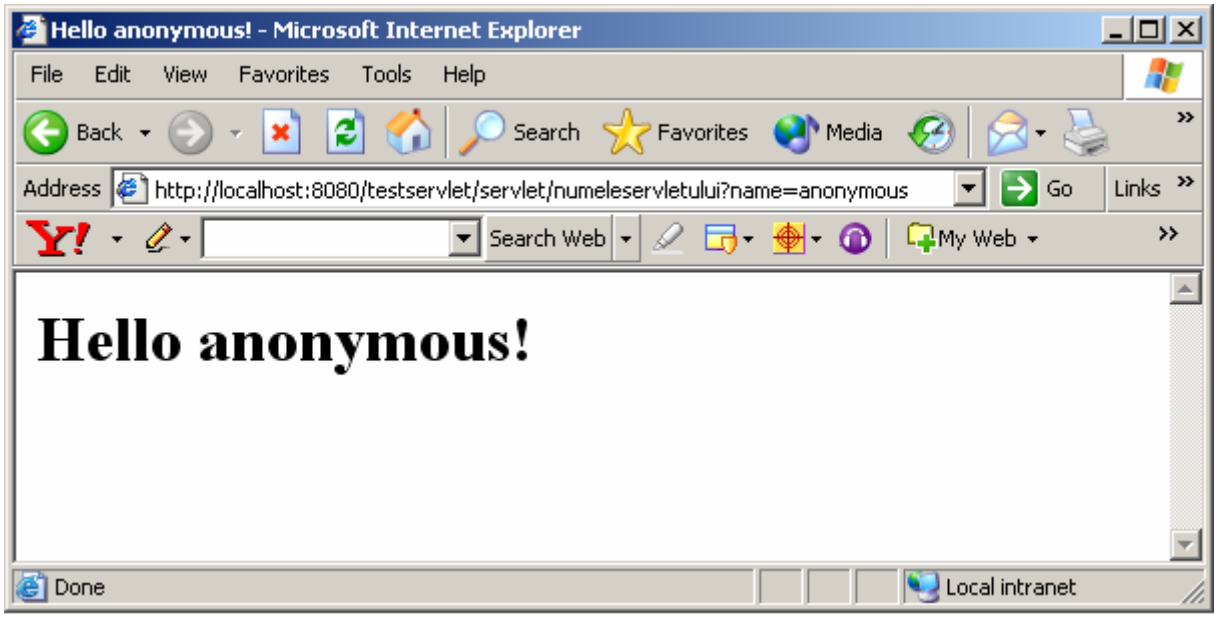
JSP.3.2. Utilizarea servlet-ului

Fisierul index.html incarcat in browser de pe server (utilizand URL-ul http://localhost:8080/testservlet/)



Dupa incarcarea formularului de acces la *servlet* (FormularAccesLaServlet.html)

Dupa apelul GET catre *servlet* (obtinut prin apasarea butonului “Say Hello”)



Continutul <u>html</u> al browserului (obtinut cu *View Source*)

```
1  <html>
2  <head>
3  <title> Hello anonymous! </title>
4  </head>
5  <body bgcolor="white">
6  <h1> Hello anonymous! </h1>
7  </body>
8  </html>
```

Distributia [testservlet.zip] (arhiva care contine instalarea *servlet-ului*).

In laborator:

Sa se instaleze si execute exemplele simple anterioare si sa se compare complexitatea crearii si instalarii fiecareia dintre cele doua solutii, JSP-ul si *servlet-ul*.