CS387 Tutorial

Building Internet Applications with Servlets and JSPs

Outline

- Java Servlets
- JSP
- ASP
- PHP

Why Build Web Pages Dynamically?

- The Web page is based on data submitted by the user
  - E.g., results page from search engines and order-confirmation pages at on-line stores
- The Web page is derived from data that changes frequently
  - E.g., a weather report or news headlines page
- The Web page uses information from databases or other server-side sources
  - E.g., an e-commerce site could use a servlet to build a Web page that lists the current price and availability of each item that is for sale.

Java Servlets

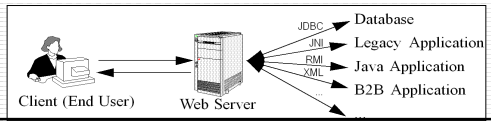
- A Java servlet is a server-side application written in Java language
- It dynamically generates HTML pages
- JVM (Java Virtual Machine) and the package javax.servlet is needed in order to run a Java Servlet
- Java servlets are very similar to CGI scripts. BUT servlets are platform independent, and much more… thread-safe, secure, portable

The Advantages of Servlets Over "Traditional" CGI

- Efficient
  - Threads instead of OS processes, one servlet copy, persistence
- Convenient
  - Lots of high-level utilities
- Powerful
  - Sharing data, pooling, persistence
- Portable
  - Run on virtually all operating systems and servers
- Secure
  - No shell escapes, no buffer overflows
- Inexpensive
  - There are plenty of free and low-cost servers.

A Servlet's Job

- Read explicit data sent by client (form data)
- Read implicit data sent by client (request headers)
- Generate the results
- Send the explicit data back to client (HTML)
- Send the implicit data to client (status codes and response headers)

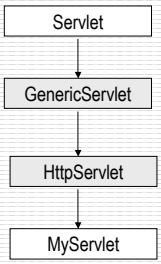


Usage of Java Servlets

- Handling HTTP client requests
- Capable of serving multiple clients concurrently
- Forwarding requests
  - Forward requests to other servers and servlets, to balance load among several servers

Architecture

- `Servlet` is the interface which all servlets will implement.
- Usually, we implement our own servlets program by extending HttpServlet class



API of Java Servlets

- Methods in the `HttpServlet` class that handle client requests take two arguments:
  - An `HttpServletRequest` object, which encapsulates the data ***from*** the client. It provides access to HTTP header data and obtain the arguments that the client sent as part of the request.
  - An `HttpServletResponse` object, which encapsulates the response ***to*** the client. It returns data to the user by Writer (plain text) or ServletOutputStream (binary).

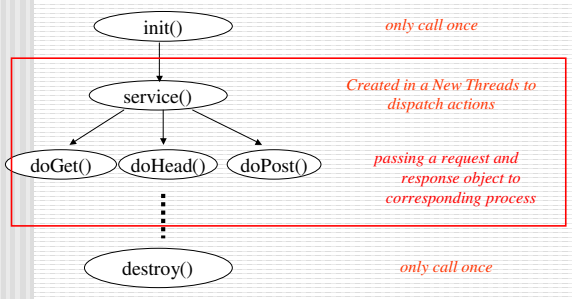
API of Java Servlets

- The methods to which the service method delegates HTTP requests include,
  - `doGet`, for handling GET, conditional GET, and HEAD requests
  - `doPost`, for handling POST requests
  - `doPut`, for handling PUT requests
  - `doDelete`, for handling DELETE requests

API of Java Servlets

- Servlets have special features available through the Servlet API classes, such as session and cookie management
- Session tracking is a mechanism that servlets use to maintain state about a series of requests from the same user across some period of time.
  - get data from the HttpSession object
- Cookies are used to access the state-information associated with the user
  - get data from the Cookie object

Servlet life cycle



Running Servlets

- **Tomcat 5.5** implements a new servlet container (called Catalina) that is based on completely new architecture. The 5.5 release implements the **Servlet 2.4** and **JSP 2.0** specifications.
  - http://tomcat.apache.org/

- Other available Servlets containers, , such as JRun by Allaire, but need $$ to use

Java Servlet Example – Hello world

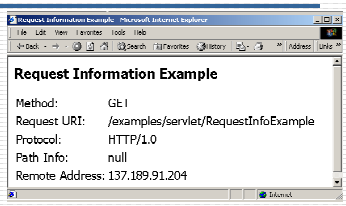
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class HelloWorld extends HttpServlet {
      public void doGet(HttpServletRequest request,  HttpServletResponse response)
throws IOException, ServletException    {
            response.setContentType("text/html");
            PrintWriter out = response.getWriter();
            out.println("<html>");
            out.println("<body>");
            out.println("<head>");
            out.println("<title>Hello World!</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Hello World!</h1>");
            out.println("</body>");
            out.println("</html>");
      }
}
```

Processing Environment Variables

- To access the Environment variable, we could use the `HttpServletRequest` object's getXXX method:
  - Examples: getMethod(), getProtocol(), getRemoteAddr() …..
- http://java.sun.com/j2ee/1.4/docs/api/javax/servlet/http/HttpServletRequest.html

HttpServletRequest environment variable example

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class RequestInfo extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)    throws IOException, ServletException    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<body>");
        out.println("<head>");
        out.println("<title>Request Information Example</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h3>Request Information Example</h3>");
        out.println("Method: " + request.getMethod());
          out.println("Request URI: " + request.getRequestURI());
          out.println("Protocol: " + request.getProtocol());
          out.println("PathInfo: " + request.getPathInfo());
          out.println("Remote Address: " + request.getRemoteAddr());
        out.println("</body>");
        out.println("</html>");
    }
    /* same operationf for doPost and doGet methods.    */
    public void doPost(HttpServletRequest request, HttpServletResponse response)    throws IOException, ServletException    {
        doGet(request, response);
    }
}
```



More examples? Install tomcat and test the provided.

JSP - JavaServer Pages

- JSP technology enables rapid development of web-based applications that are platform independent
- JSP technology separates the user interface from content generation enabling designers to change the overall page layout without altering the underlying dynamic content
- JSP technology lets you mix regular, static HTML with dynamically-generated HTML
- JavaServer Pages technology is an extension of the Java Servlet technology.

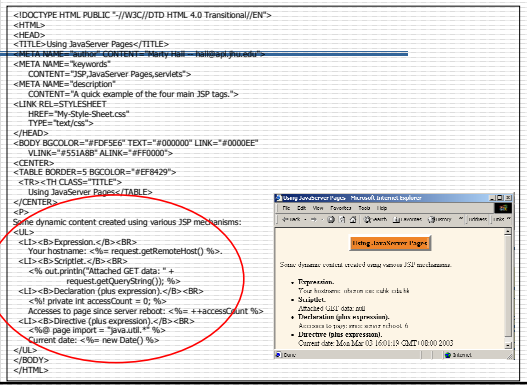
Writing a JSP

- Give your file a .jsp extension, and typically install it in any place you could place a normal Web page
- Enclose the code for the dynamic parts in special tags, most of which start with "<%" and end with "%>".

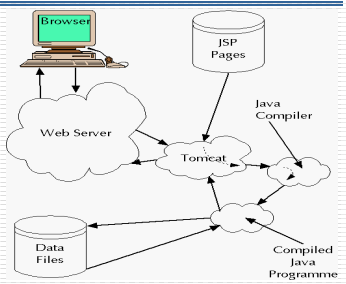
JSP basic syntax

JSP Element	Syntax	Meaning	Example
JSP Expression	<%= expression %>	Expression is evaluated and placed in output.	Your hostname: <%= request.getRemoteHost() %>
JSP Scriptlet	<% code %>	Code is inserted in service method.	<% out.println("Hello"); %>
JSP Declaration	<%! code %>	Code is inserted in body of servlet class, outside of service method.	<%! private int accessCount = 0; %>
JSP Comment	<%-- comment --%>	Comment; ignored when JSP page is translated into servlet.	<%-- it is a comment--%>

JSP Example



System Architecture



ASP - Active Server Pages

- File ended with .asp extension
- Developed by Micro$oft
- Using VBScript or Jscript
- Usually work with Micro$oft Internet Information Server (IIS) in WinNT/2K/XP or Personal Web Server (PWS) in Win 9X

ASP

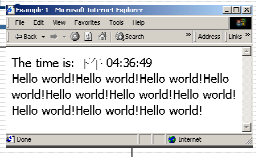
- Web pages that contain server-side scripts in addition to HTML tags and text
- When a browser requests an ASP file, IIS passes the request to the ASP engine. The ASP engine reads the ASP file and executes the scripts in the file. Finally, the ASP file is returned to the browser as plain HTML

Writing ASP

- You can include the ASP statement inside <%….%> in your web page.
- <%@Language=”VBScript”%> tells the browser the default scripting language is VBScript.
- <% option explicit %> ensures all variables must be declared before their use.

ASP example 1:

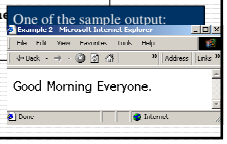
```
<%@ Language=VBScript %>
<html>
<head>
      <title>Example 1</title>
</head>
<body>
      <% FirstVar = "Hello world!" %>
      The time is: <%=time%> <BR>
      <%FOR i=1 TO 10%>
            <%=FirstVar%>
      <%NEXT%>
</body>
</html>
```



ASP example 2:

```
<%@ Language=VBScript %>
<html>
<head>
      <title>Example 2</title>
</head>
<body>
      <%IF Hour(time)>18 OR Hour(time)<4
THEN%>
            Good Night Everyone.
      <%ELSE%> Good Morning Everyone
IF%>
</body>
</html>
```

One of the sample output:



JSP vs ASP

- For speed, ASP is much faster than the corresponding JSPs
  - http://www.devhood.com/Tools/tool_details.aspx?tool_id=793
- For portability and security, JSP would be better.

PHP: Hypertext Preprocessor

- Similar to JSP and ASP
- Perl-liked syntax
- The strongest and most significant feature in PHP is its support for a wide range of databases. Writing a database-enabled web page is incredibly simple. The following databases are currently supported:

Adabas D	Ingres	Oracle (OCI7 and OCI8)	
dBase	InterBase	Ovrimos Informix	Empress
FrontBase	PostgreSQL	ODBC	FilePro
mSQL	Direct	MS-SQL	dbm
Solid	Hyperwave	Sybase	IBM DB2
MySQL	Velocis		

PHP syntax

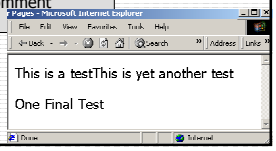
```
<?
PHP Code In Here
?>

<?php
PHP Code In Here
php?>

<script language="php">
PHP Code In Here
</script>
```

PHP example

```
<?php echo "This is a test";
        // This is a one-line c++ style comment
        /* This is a multi line comment yet
another line of comment */
                echo "This is yet another
test</p>";
                echo "One Final Test";
        # This is shell-style style comment
?>
```



Conclusion

- Servlets and CGI are programs which are run in server side. The outputs will be sent to the user browsers for display.
- JSP, ASP and PHP are scripts embedded in HTML file, which can generate dynamic HTML contents.
- Servlets and CGI are more powerful in general, while JSP, ASP and PHP are more convenientt to use and write.