16. JSP II

충남대학교 컴퓨터공학과
데이타베이스시스템 연구실

Client Request

- **When a browser requests for a web page,**
  - it sends lot of information to the web server which can not be read directly because this information travel as a part of header of HTTP request.
  - Important header information which comes from browser side *Try it!*

- **The HttpServeltRequest Object**
  - request object is an instance of a javax.servlet.http.HttpSevletRequest object.
  - request object provides methods to get HTTP header information including from data, cookies, HTTP methods etc.
  - These method are available with HttpServletRequest object *Try it!*

Client Request (cont'd)

- HTTP Header Request Example *Try it!*

```
<%@ page import="java.io.*,java.util.*" %>
<html>
<head>
<title>HTTP Header Request Example</title>
</head>
<body>
<center>
<h2>HTTP Header Request Example</h2>
<table width="100%" border="1" align="center">
<tr bgcolor="#949494">
<th>Header Name</th><th>Header Value(s)</th>
</tr>
<%
   Enumeration headerNames = request.getHeaderNames();
   while(headerNames.hasMoreElements()) {
      String paramName = (String)headerNames.nextElement();
      out.print("<tr><td>" + paramName + "</td>\n");
      String paramValue = request.getHeader(paramName);
      out.println("<td> " + paramValue + "</td></tr>\n");
   }
%>
</table>
</center>
</body>
</html>
```

Server Response

- When a Web server responds to a HTTP request to the browser

```
HTTP/1.1 200 OK
Content-Type: text/html
Header2: ...
...
HeaderN: ...
  (Blank Line)
<!doctype ...>
<html>
<head>...</head>
<body>
...
</body>
</html>
```

- A summary of the most useful HTTP 1.1 response headers which go back to the browser from web server side *Try it!*

Server Response (cont'd)

- **The HttpServeltResponse Object**
  - response object is an instance of a javax.servlet.http.HttpSevletResponse object.
  - request object defines the interface that deal with creating new HTTP headers. Through this object the JSP programmer can add new cookies or date stamps, HTTP status codes etc
  - These method are available with HttpServletResponse object *Try it!*

Server Response (cont'd)

- HTTP Header Response Example *Try it!*

```
<%@ page import="java.io.*,java.util.*" %>
<html>
<head>
<title>Auto Refresh Header Example</title>
</head>
<body>
<center>
<h2>Auto Refresh Header Example</h2>
<%
   // Set refresh, autoload time as 5 seconds
   response.setIntHeader("Refresh", 5);
   // Get current time
   Calendar calendar = new GregorianCalendar();
   String am_pm;
   int hour = calendar.get(Calendar.HOUR);
   int minute = calendar.get(Calendar.MINUTE);
   int second = calendar.get(Calendar.SECOND);
   if(calendar.get(Calendar.AM_PM) == 0)
      am_pm = "AM";
   else
      am_pm = "PM";
   String CT = hour+":"+ minute +":"+ second +" "+ am_pm;
   out.println("Current Time is: " + CT + "\n");
%>
</center>
</body>
</html>
```

HTTP Status Codes

- *A list of HTTP status codes and associated messages*

- **Methods to Set HTTP Status Code**
  - Can be used to set HTTP Status Code in your servlet program
  - Available with *HTTPServletResponse* object

S.N.	Method & Description
1	**public void setStatus (int statusCode)** This method sets an arbitrary status code. The setStatus method takes an int (the status code) as an argument. If your response includes a special status code and a document, be sure to call setStatus before actually returning any of the content with the *PrintWriter*.
2	**public void sendRedirect(String url)** This method generates a 302 response along with a *Location* header giving the URL of the new document.
3	**public void sendError(int code, String message)** This method sends a status code (usually 404) along with a short message that is automatically formatted inside an HTML document and sent to the client.

HTTP Status Codes (cont'd)

- **HTTP Status Code Example**
  - http://168.188.128.44:8080/lecture/16-3.jsp

```
<html>
<head>
<title>Setting HTTP Status Code</title>
</head>
<body>
<%
   // Set error code and reason.
   response.sendError(407, "Need authentication!!!" );
%>
</body>
</html>
```

Form Processing

- **GET method (default method)**
  - Sends the encoded user information appended to the page request.

```
http://www.test.com/hello?key1=value1&key2=value2
```

  - This information is passed using QUERY_STRING header.
  - Accessible through QUERY_STRING environment variable which can be handled using getQueryString( ) and getParameter( ) methods of request object.

Form Processing (cont'd)

- **POST method**
  - A generally more reliable method of passing information to a backend program
  - Packages the information in exactly the same way as GET methods, but instead of sending it as a text string after a ? In the URL it sends it as a separate message.
  - getParameter( ) method to read simple parameters
  - getInputStream( ) method to read binary data stream coming from the client.

Form Processing (cont'd)

- **Reading Form Data using JSP**
  - getParameter( )
    - You call request.getParameter( ) method to get the value of a form parameter
  - getParameterValues( )
    - Call this method if the parameter appears more than once and returns multiple values, for example checkbox
  - getParameterNames( )
    - Call this method if you want a complete list of all parameters in the current request
  - getInputStream( )
    - Call this method to read binary data stream coming from the clinet

Form Processing (cont'd)

- **Ex1 : GET Method Example** *Try it!*

```
<html>
<body>
<form action="main.jsp" method="GET">
First Name: <input type="text" name="first_name">
<br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

Form Processing (cont'd)

- main.jsp

```
<html>
<head>
<title>Using GET Method to Read Form Data</title>
</head>
<body>
<center>
<h1>Using GET Method to Read Form Data</h1>
<ul>
<li><p><b>First Name:</b>
   <%= request.getParameter("first_name")%>
</p></li>
<li><p><b>Last  Name:</b>
   <%= request.getParameter("last_name")%>
</p></li>
</ul>
</body>
</html>
```

Form Processing (cont'd)

- **Ex 2: POST Method Example** *Try it!*

```
<html>
<body>
<form action="main.jsp" method="POST">
First Name: <input type="text" name="first_name">
<br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

Form Processing (cont'd)

- **Ex3 : Passing Checkbox Data to JSP Program** *Try it!*

```
<html>
<body>
<form action="main.jsp" method="POST" target="_blank">
<input type="checkbox" name="maths" checked="checked" /> Maths
<input type="checkbox" name="physics"  /> Physics
<input type="checkbox" name="chemistry" checked="checked" />
                                                Chemistry
<input type="submit" value="Select Subject" />
</form>
</body>
</html>
```

Form Processing (cont'd)

- main.jsp for Ex3

```
<html>
<head>
<title>Reading Checkbox Data</title>
</head>
<body>
<center>
<h1>Reading Checkbox Data</h1>
<ul>
<li><p><b>Maths Flag:</b>
   <%= request.getParameter("maths")%>
</p></li>
<li><p><b>Physics Flag:</b>
   <%= request.getParameter("physics")%>
</p></li>
<li><p><b>Chemistry Flag:</b>
   <%= request.getParameter("chemistry")%>
</p></li>
</ul>
</body>
</html>
```

Form Processing (cont'd)

- **Ex 4 : Reading All Form Parameters** *Try it!*

```
<html>
<body>
<form action="main.jsp" method="POST" target="_blank">
<input type="checkbox" name="maths" checked="checked" /> Maths
<input type="checkbox" name="physics"  /> Physics
<input type="checkbox" name="chemistry" checked="checked" /> Chem
<input type="submit" value="Select Subject" />
</form>
</body>
</html>
```

Form Processing (cont'd)

- main.jsp for Ex4

```
<%@ page import="java.io.*,java.util.*" %>
<html>
<head>
<title>HTTP Header Request Example</title>
</head>
<body>
<center>
<h2>HTTP Header Request Example</h2>
<table width="100%" border="1" align="center">
<tr bgcolor="#949494">
<th>Param Name</th><th>Param Value(s)</th>
</tr>
<%
   Enumeration paramNames = request.getParameterNames();

   while(paramNames.hasMoreElements()) {
      String paramName = (String)paramNames.nextElement();
      out.print("<tr><td>" + paramName + "</td>\n");
      String paramValue = request.getHeader(paramName);
      out.println("<td> " + paramValue + "</td></tr>\n");
   }
%>
</table>
</center>
</body>
</html>
```

Session Tracking

- **The session Object**
  - JSP makes use of servlet provided HttpSession Interface which provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user.
  - By default, JSP have session tracking enabled and a new HttpSession object is instantiated for each new client automatically.
  - JSP engine exposes the HttpSession object to the JSP author through the implicit **session** object

Session Tracking (cont'd): *Try it!*

- Important methods available through session object

S.N.	Method & Description
1	**public Object getAttribute(String name)** This method returns the object bound with the specified name in this session, or null if no object is bound under the name.
2	**public Enumeration getAttributeNames()** This method returns an Enumeration of String objects containing the names of all the objects bound to this session.
3	**public long getCreationTime()** This method returns the time when this session was created, measured in milliseconds since midnight January 1, 1970 GMT.
4	**public String getId()** This method returns a string containing the unique identifier assigned to this session.
5	**public long getLastAccessedTime()** This method returns the last time the client sent a request associated with this session, as the number of milliseconds since midnight January 1, 1970 GMT.
6	**public int getMaxInactiveInterval()** This method returns the maximum time interval, in seconds, that the servlet container will keep this session open between client accesses.
7	**public void invalidate()** This method invalidates this session and unbinds any objects bound to it.
8	**public boolean isNew(** This method returns true if the client does not yet know about the session or if the client chooses not to join the session.
9	**public void removeAttribute(String name)** This method removes the object bound with the specified name from this session.
10	**public void setAttribute(String name, Object value)** This method binds an object to this session, using the name specified.
11	**public void setMaxInactiveInterval(int interval)** This method specifies the time, in seconds, between client requests before the servlet container will invalidate this session.

Session Tracking (cont'd)

- **Deleting Session Data**
  - Remove a particular attribute
    - You can call *public void removeAttribute(String name)* method to delete the value associated with a particular key.
  - Delete the whole session
    - You can call *public void invalidate()* method to discard an entire session.
  - Setting Session timeout
    - You can call *public void setMaxInactiveInterval(int interval)* method to set the timeout for a session individually.
  - Log the user out
    - The servers that support servlets 2.4, you can call **logout** to log the client out of the Web server and invalidate all sessions belonging to all the users.

Handling Date

- Once you have a Date object available, you can call any of the following support methods to play with dates *Try it!*

- Getting Current Date & Time *Try it!*

```
<%@ page import="java.io.*,java.util.*, javax.servlet.*" %>
<html>
<head>
<title>Display Current Date & Time</title>
</head>
<body>
<center>
<h1>Display Current Date & Time</h1>
</center>
<%
   Date date = new Date();
   out.print( "<h2 align=\"center\">" +date.toString()+"</h2>");
%>
</body>
</html>
```

Handling Date (cont'd)

- **Date Formatting using SimpleDateFormat** *Try it!*
  - *Simple DateFormat format codes*

```
<%@ page import="java.io.*,java.util.*" %>
<%@ page import="javax.servlet.*,java.text.*" %>
<html>
<head>
<title>Display Current Date & Time</title>
</head>
<body>
<center>
<h1>Display Current Date & Time</h1>
</center>
<%
   Date dNow = new Date( );
   SimpleDateFormat ft =
   new SimpleDateFormat ("E yyyy.MM.dd 'at' hh:mm:ss a zzz");
   out.print( "<h2 align=\"center\">" + ft.format(dNow) + "</h2>");
%>
</body>
</html>
```

Page Redirecting

- **Page redirection**
  - Is generally used when a document moves to a new location and we need to send the client to this new location.
  - Using method **sendRedirect()** of response object

```
public void response.sendRedirect(String location)
throws IOException
```

  - This method sends back the response to the browser along with the status code and new page location.
    - You can also use setStatus( ) and setHeader( ) methods together to achieve the same redirection

```
....
String site = "http://www.newpage.com" ;
response.setStatus(response.SC_MOVED_TEMPORARILY);
response.setHeader("Location", site);
....
```

Page Redirecting (cont'd)

- Example *Try it!*

```
<%@ page import="java.io.*,java.util.*" %>
<html>
<head>
<title>Page Redirection</title>
</head>
<body>
<center>
<h1>Page Redirection</h1>
</center>
<%
   // New location to be redirected
   String site = new String("http://www.photofuntoos.com");
   response.setStatus(response.SC_MOVED_TEMPORARILY);
   response.setHeader("Location", site);
%>
</body>
</html>
```

Auto Refresh

- **Consider a webpage**
  - Is displaying live game score or stock market status or currency exchange ration. For all such type of pages, you would need to refresh your web page regularly using refresh or reload button with your browser.

- **You can make a webpage in such a way that it would refresh automatically after a given interval**
  - Using method **setIntHeader( )** of response object.

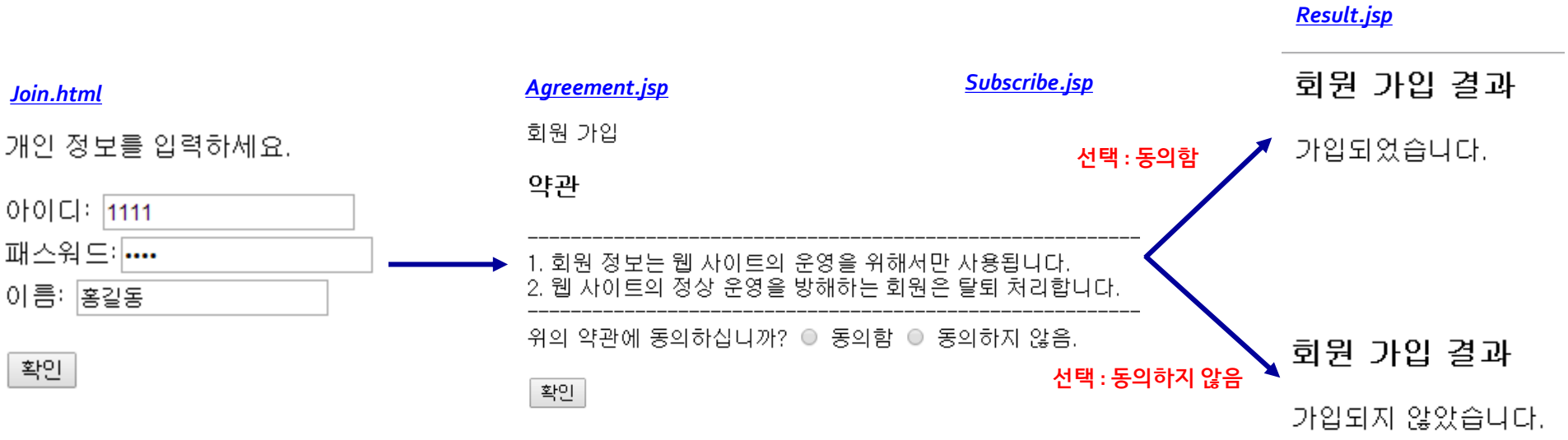
```
public void setIntHeader(String header, int headerValue)
```

Auto Refresh (cont'd)

- Example *Try it!*

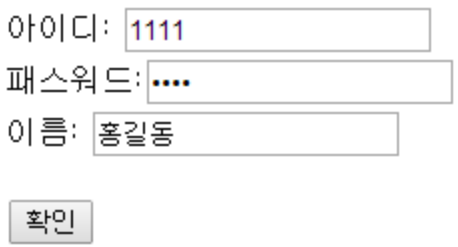
```
<%@ page import="java.io.*,java.util.*" %>
<html>
<head>
<title>Auto Refresh Header Example</title>
</head>
<body>
<center>
<h2>Auto Refresh Header Example</h2>
<%
   // Set refresh, autoload time as 5 seconds
   response.setIntHeader("Refresh", 5);
   // Get current time
   Calendar calendar = new GregorianCalendar();
   String am_pm;
   int hour = calendar.get(Calendar.HOUR);
   int minute = calendar.get(Calendar.MINUTE);
   int second = calendar.get(Calendar.SECOND);
   if(calendar.get(Calendar.AM_PM) == 0)
      am_pm = "AM";
   else
      am_pm = "PM";
   String CT = hour+":"+ minute +":"+ second +" "+ am_pm;
   out.println("Crrent Time: " + CT + "\n");
%>
</center>
</body>
</html>
```

회원 가입 예제 *Try it!*



예제 설명 : Join.html

- 회원 가입을 위한 화면
  - 아이디
  - 패스워드
  - 이름

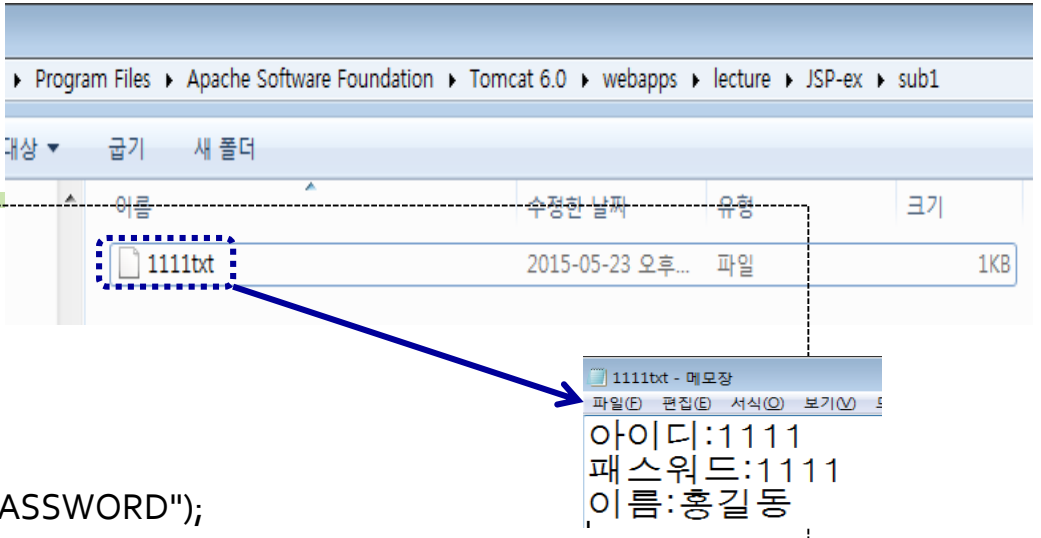



```
<HTML>

  <HEAD>
    <META charset="UTF-8"/>
    <TITLE>JSP 전체 예제 : 회원 가입</TITLE>
  </HEAD>

  <BODY>
    개인 정보를 입력하세요.<BR><BR>

   <FORM ACTION=Agreement.jsp  METHOD=POST>
      아이디:  <INPUT TYPE=TEXT NAME=ID><BR>
      패스워드:<INPUT TYPE=PASSWORD NAME=PASSWORD><BR>
      이름:   <INPUT TYPE=TEXT NAME=NAME><BR><BR>
      <INPUT TYPE=SUBMIT VALUE='확인'>
    </FORM>

  </BODY>
</HTML>
```

예제 설명 : Aggrement.jsp

```
<%@page contentType="text/html; charset=UTF-8"%>
<%
        request.setCharacterEncoding("UTF-8");
        String id = request.getParameter("ID");
        String password = request.getParameter("PASSWORD");
        String name = request.getParameter("NAME");
        session.setAttribute("ID", id);
        session.setAttribute("PASSWORD", password);
        session.setAttribute("NAME", name);
%>
<HTML>
    <HEAD<TITLE>회원 가입</TITLE></HEAD>
     <BODY>
         <H3>약관</H3>
         ----------------------------------------------------------------------<BR>
         1. 회원 정보는 웹 사이트의 운영을 위해서만 사용됩니다.<BR>
         2. 웹 사이트의 정상 운영을 방해하는 회원은 탈퇴 처리합니다.<BR>
         ----------------------------------------------------------------------<BR>
         <FORM ACTION=Subscribe.jsp METHOD=POST>   위의 약관에 동의하십니까?
         <INPUT TYPE=RADIO NAME=AGREE VALUE=YES> 동의함
         <INPUT TYPE=RADIO NAME=AGREE VALUE=NO> 동의하지 않음.<BR><BR>
        <INPUT TYPE=SUBMIT VALUE='확인'>
        </FORM> </BODY>
</HTML>
```

예제 설명 : Subscribe.jsp




```
<%@page contentType="text/html; charset=UTF-8"%>
<%@page import="java.io.*"%>
<%  String agree = request.getParameter("AGREE");
        String result = null;
        if (agree.equals("YES")) {
              String id=(String) session.getAttribute("ID");
              String password=(String) session.getAttribute("PASSWORD");
              String name=(String) session.getAttribute("NAME");
              PrintWriter writer=null;
         try {
              String filePath = application.getRealPath("/JSP-ex/sub1/"+id+"txt");
             writer=new PrintWriter(filePath);
             writer.println("아이디:" + id);
             writer.println("패스워드:" + password);
             writer.println("이름:" + name);
             result="SUCCESS"; }
         catch (IOException ioe) { result="FAIL"; }
        finally {
              try { writer.close();}
             catch (Exception e) { ...}}}
       else { result="FAIL";  }
       session.invalidate();
       response.sendRedirect("Result.jsp?RESULT="+result);
%>
```

예제 설명 : Result.jsp

```
<%@page contentType="text/html; charset=UTF-8"%>
<% String result=request.getParameter("RESULT");%>
<HTML>
    <HEAD><TITLE>회원 가입</TITLE></HEAD>
    <BODY>
        <H3>회원 가입 결과</H3>
        <%
            if(result.equals("SUCCESS"))
                out.println("가입되었습니다.");
            else
                out.println("가입되지 않았습니다.");
        %>
    </BODY>
</HTML>
```