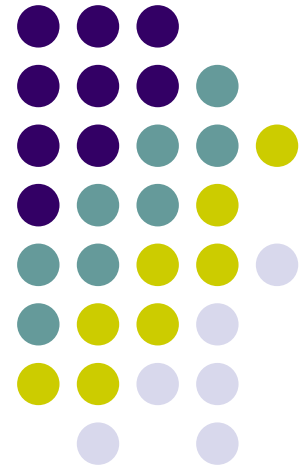


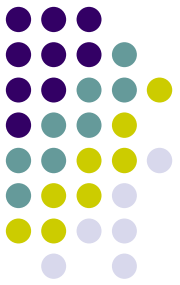
# Servlets and JSP

---

CMPUT 410 – lab

Rimon Mikhaiel

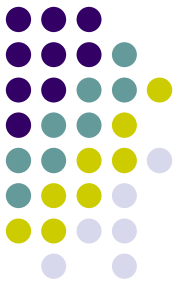




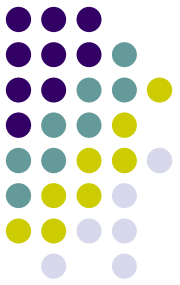
# Agenda

- Static vs Dynamic Pages
- Servlets and JSP
- A Servlet Example
- Your First JSP
  - Expressions
  - Scriptlets
  - JSP Directives
- Form Processing: example
- References

# Static vs Dynamic Pages



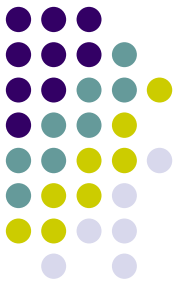
- Static Pages: HTML only.
- Dynamic Pages:
  - Client Side:
    - Processing takes place at the client side.
    - HTML is manipulated in combination with Javascript or Java Applets.
      - In case of Java applets, a client needs a virtual machine.
    - Similar requests always result in similar responses, and then it is up to the client side to process it.
  - Server Side:
    - Processing takes places at the server side producing the HTML to be displayed to the user.
    - The user does not need to install any kind of virtual machine on his side.
    - Servlets, JSP, PHP, CGI, and ASP.
    - Similar request may have different responses based on the server side logic.



# Servlets and JSP

- A *servlet* is a Java class that provides special server side service, while a Java Server Page (JSP) is an HTML page with embedded code.
- All Java Server Pages eventually become *servlets* before executing.

# A Servlet Example



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

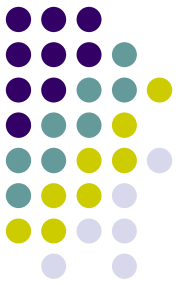
public class SomeServlet extends HttpServlet
{
    public void doGet(        HttpServletRequest request,
                        HttpServletResponse response)
        throws ServletException, IOException
    {

        // Use "request" to read incoming HTTP headers (e.g. cookies)
        // and HTML form data (e.g. data the user entered and submitted)

        // Use "response" to specify the HTTP response line and headers
        // (e.g. specifying the content type, setting cookies).

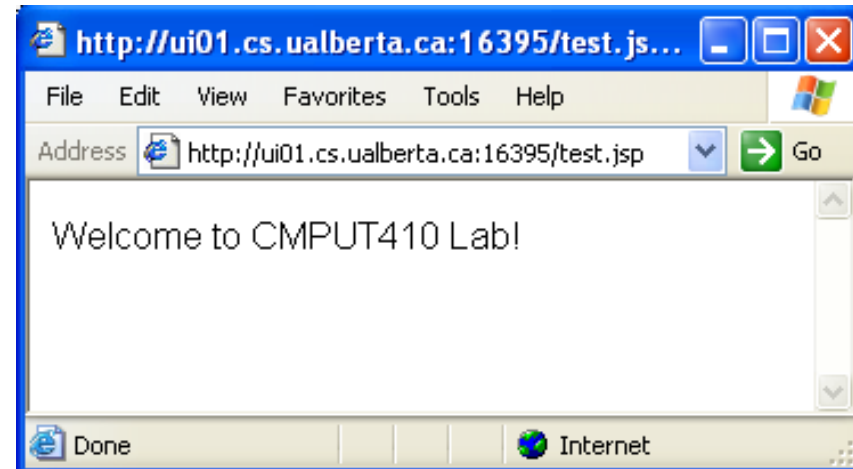
        PrintWriter out = response.getWriter();
        // Use "out" to send content to browser
        out.println("<h1>Hello World</h1>");
    }
}
```

# Your First JSP

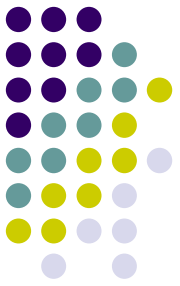


```
<HTML>
  <BODY>
    Welcome to CMPUT410
    Lab!
  </BODY>
</HTML>
```

- Store the file in the your JSP directory:  
\$HOME/catalina/webapps/test.jsp
- Load the new file, with the ".jsp" extension, in your browser.
- `http://machine_name.cs.ualberta.ca:port_number/proj1/welcome.jsp`



# Adding Dynamic Content via Expressions

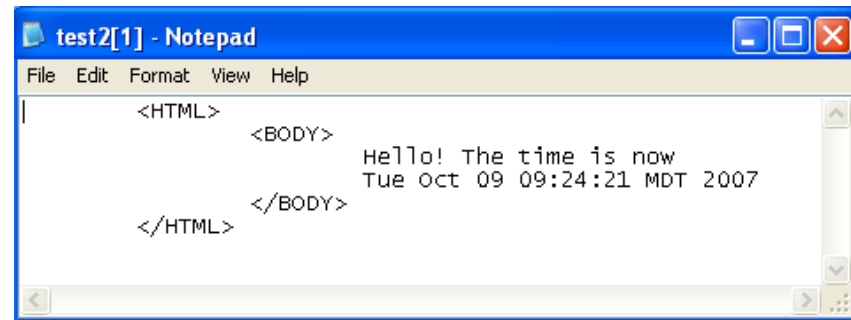
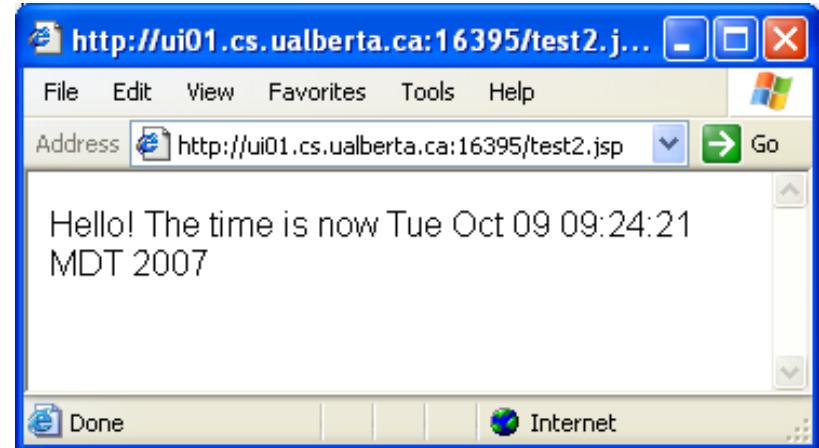


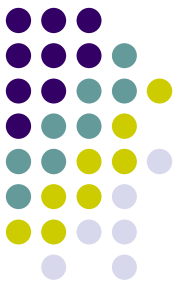
- What makes JSP useful is the ability to embed Java.
- Embed Java expressions in JSP pages by putting them between the Expression tag: `<%= and %>`.

```
<HTML>
  <BODY>
    Hello! The time is now
    <%= new java.util.Date() %>

  </BODY>
</HTML>
```

- Load this page, and have a look on the produced HTML source.
- A response is produced by replacing each jsp block by its output, if exists.

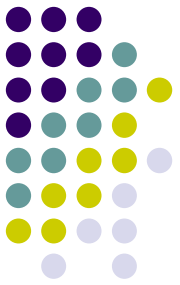




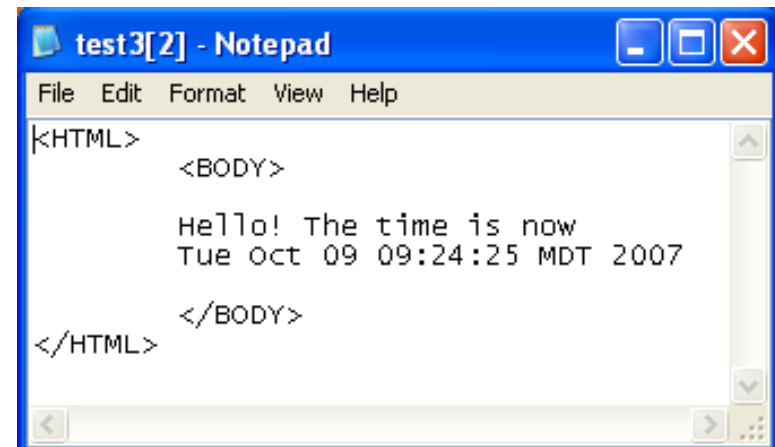
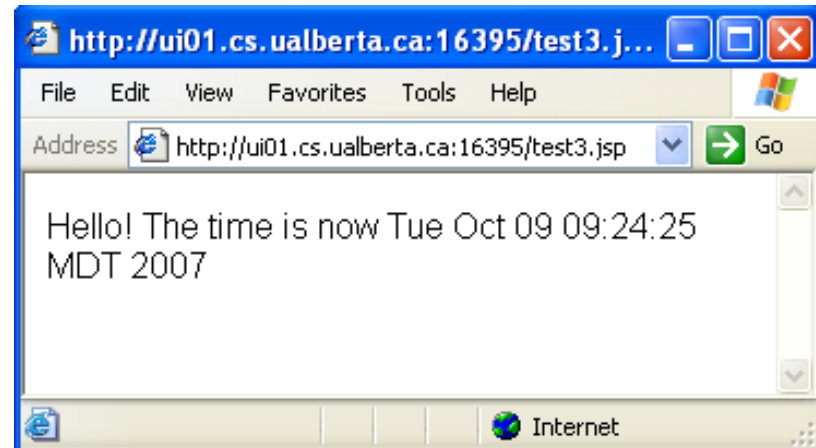
# Scriptlets

- *Scriptlet*: blocks of Java code inside the JSP page.
- You do this by placing your Java code between `<%` and `%>` characters (just like expressions, but without the `=` sign at the start of the sequence.)
- A *scriptlet* does not generate HTML, however, an object called “out” can be used to print HTML to the response.

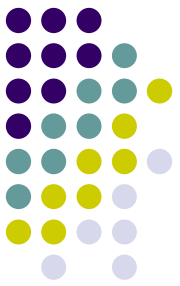
# A Scriptlet example



```
<HTML>
  <BODY>
    <%
      // This scriptlet declares and
      initializes "date"
      java.util.Date date = new
      java.util.Date();
    %>
    Hello! The time is now
    <%
      // This scriptlet generates HTML output
      out.println( String.valueOf( date ));
    %>
  </BODY>
</HTML>
```



# Mixing Scriptlets and HTML

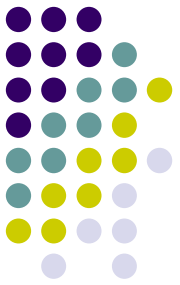


```
<HTML>
  <BODY>
    <TABLE BORDER=2>
      <% for(int i = 0; i<5; i++ )
        {
          %>
            <TR>
              <TD>Number</TD>
              <TD><%= i+1 %></TD>
            </TR>
          <%
        }
      %>
    </TABLE>
  </BODY>
</HTML>
```

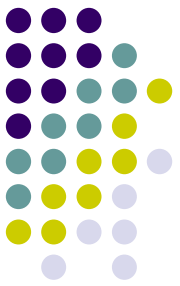
Number	1
Number	2
Number	3
Number	4
Number	5

```
<HTML>
  <BODY>
    <TABLE BORDER=2>
      <TR>
        <TD>Number</TD>
        <TD>1</TD>
      </TR>
      <TR>
        <TD>Number</TD>
        <TD>2</TD>
      </TR>
      <TR>
        <TD>Number</TD>
        <TD>3</TD>
      </TR>
      <TR>
        <TD>Number</TD>
        <TD>4</TD>
      </TR>
      <TR>
        <TD>Number</TD>
        <TD>5</TD>
      </TR>
    </TABLE>
  </BODY>
</HTML>
```

# JSP Directives

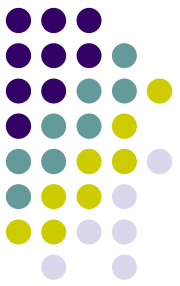


```
<%@ page import="java.util.*" %>
<HTML>
  <BODY>
    <%
      java.util.Date date = new Date();
    %>
    Hello! The time is now <%= date %>
  </BODY>
</HTML>
```



# JSP Directives (Cont.)

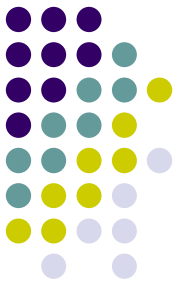
- A JSP directive gives special information about the page to the JSP Engine.
- Directive tag: `<%@ directive ... %>`
- There are three main types of directives:
  - page - processing information for this page.
    - List of imported packages:  
`<%@ page import="java.util.* , java.io.*" %>`
  - Include - files to be included.  
`<%@ include file="another.jsp"%>`
  - Tag library - tag library to be used in this page.



# JSP Declarations

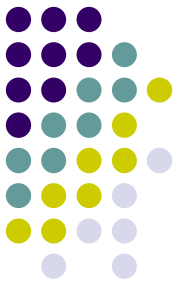
- For a JSP, all the code within the `<% %>` tags and all the expressions within the `<%= %>` tags belong to one big Java method in the generated *servlet*.
- A JSP Declaration is used to enclosed any declarations that belong outside the big method that generates the page.
- Declaration tag: `<%! ... %>`

# JSP Declarations: Example



```
<HTML>
  <BODY>
    <%!
      String name;
      String generateWelcomeMessage()
      {
          return "Welcome, "+name+".";
      }
    %>
    <%
      name="Test User";
    %>
    .
    .
    .
    <%= generateWelcomeMessage()%>
  </BODY>
</HTML>
```

# Form Processing: example



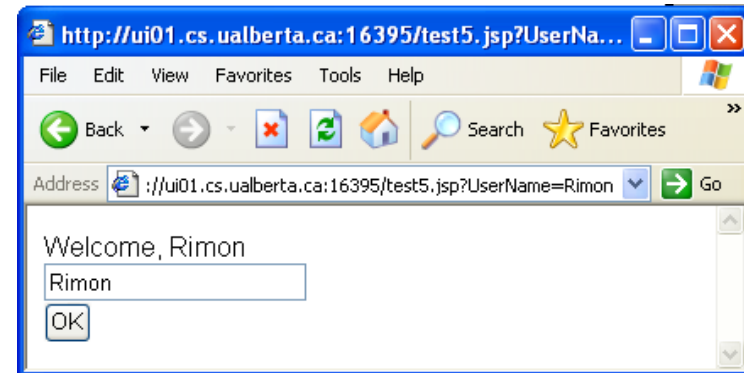
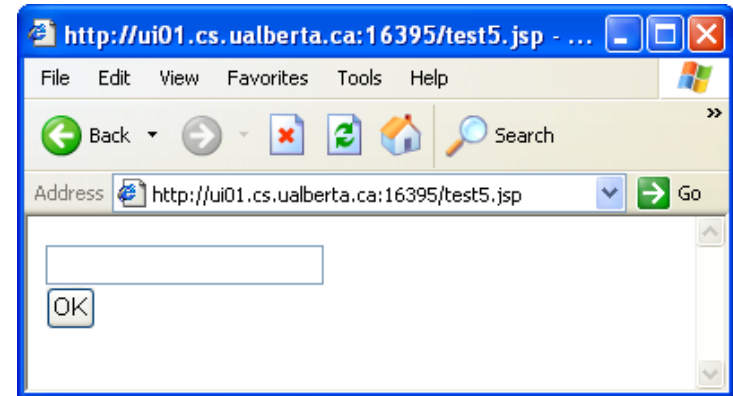
```
<HTML>
<BODY>
  <FORM>
  <%
    String name=request.getParameter("UserName");
  %>

  <%= (name.equals("")) ? "" : "Welcome, "+name%>
  <br>

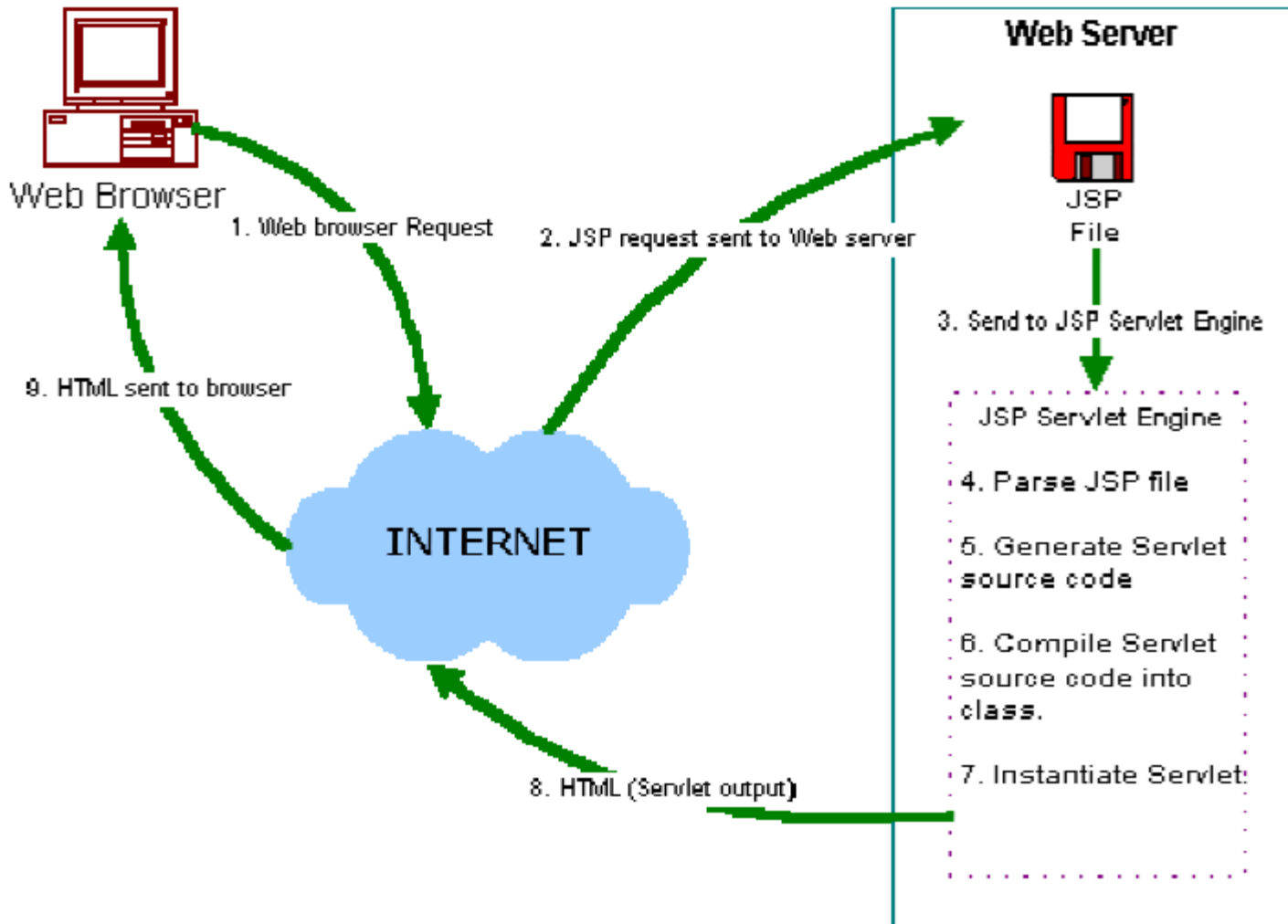
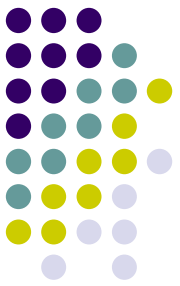
  <input type="text" name="UserName"
    value="<%=request.getParameter("UserName")%>">
  <br>

  <input type="submit" value="OK">

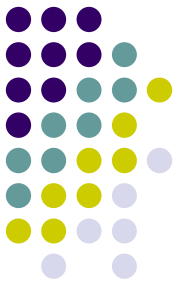
</FORM>
</BODY>
</HTML>
```



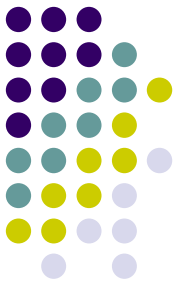
# JSP Life Cycle



# Pre-defined Objects for JSPs



- There are several objects that are automatically available in JSP:
  - Request: to get parameters values, names and information about the requesting party.
  - Response: to redirect to another page, to set content-type, etc.
  - Out: to produce displayable output.
  - Session: to store cross-page values like user-id, login-status, etc.



# References

- **Specifications and API**
  - [JSP Specifications](#)
  - [Servlet and JavaServer Pages API Documentation](#)
- **Tutorials:**
  - [JSP Specifications](#)
  - [JSP tutorial](#)
  - [Servlets and JavaServer Pages \(JSP\) 1.0: A Tutorial](#)
  - [Servlets Fundamentals](#)
- **Reference Cards:**
  - [Allaire JRun](#)
  - [JSP Syntax 1.1](#)