Web ApplicationJava Server Pages (JSP)

- What is MVC
- Why MVC

- JSP scripting elements
    - **Expressions**

      ```
      <jsp:expression>
         expression
      </jsp:expression>
      <%= "welcome to jsp" %>
      ```

    - **Scriptlets**

      ```
      <jsp:scriptlet>
         code fragment
      </jsp:scriptlet>

      <% out.print("welcome to jsp"); %>
      ```

        - **JSP - Implicit Objects**
            - **request** - This is the HttpServletRequest object associated with the request.
                - *request.getParameter("testParam");*
            - **response** - This is the HttpServletResponse object associated with the response to the client
            - **out** - This is the PrintWriter object used to send output to the client.
            - **session** - This is the HttpSession object associated with the request.
                - *session.getId();*
                - *session.setMaxInactiveInterval(10);*
            - Application , config , pageContext , Page , Exception

- **Declarations**

  ```
  <jsp:declaration>
     code fragment
  </jsp:declaration>
  ```

  ```
  <%! int data=50; %>
  <%= "Value of the variable is:"+ data %>
  ```

- Directive
  - *Which classes are imported*
  - *What class the servlet extends*
  - *What MIME type is generated*
  - *How multithreading is handled*

  - **page directive**

    Import

    - <%@ page **import**="java.util.Date" %>

      Today is: <%= **new** Date() %>

    contentType

    - <%@ page contentType=application/msword %>

    isELIgnored

    - <%@ page isELIgnored="true" %>

    extends , info , buffer , language , isThreadSafe , autoFlush , session , pageEncoding , errorPage , isErrorPage

- **include directive (Difference with Example)**

  Include directive includes the file at translation time (the phase of JSP life cycle where the JSP gets converted into the equivalent servlet) whereas the include action includes the file at runtime

  - `<%@ include file="display.jsp" %>`
  - `<jsp:include page="display.jsp" />`

- JSTL
- `<%@ taglib uri="cwp-taglib.tld" prefix="cwp" %>`

```
<cwp:repeat reps="10">
  <cwp:if>
    <cwp:condition>
      <%= Math.random() < 0.5 %>
    </cwp:condition>
    <cwp:then><B>Heads</B><BR></cwp:then>
    <cwp:else><B>Tails</B><BR></cwp:else>
  </cwp:if>
</cwp:repeat>
```

- JSP Actions
    - Jsp:**include** - Includes a file at the time the page is requested.

      ```
      <jsp:include page="display.jsp"  />
      ```

    - jsp:**useBean** - Finds or instantiates a JavaBean.

      ```
      <jsp:useBean id="name" class="package.Class" scope="request
      />
      ```

        - request
        - session
        - application
        - page
    - jsp:**setProperty** - Sets the property of a JavaBean.

      ```
      <jsp:setProperty name="book1" property="title" value="ABC" />

      <jsp:setProperty name="book1" property="title" param="title" />

      <jsp:setProperty name="book1" property="*" />
      ```

    - jsp:**getProperty** - Inserts the property of a JavaBean into the output.

      ```
      <jsp:getProperty name="book1" property="title" />
      ```

    - Jsp:**forward** - Forwards the requester to a new page.

      ```
      <jsp:forward page="another.jsp">

              <jsp:param name="callingPage" value="current.jsp">

      </jsp:forward>
      ```

    - Jsp:plugin - Generates browser-specific code that makes an OBJECT or EMBED tag for the Java plugin.
    - Jsp:element - Defines XML elements dynamically.
    - Jsp:attribute - Defines dynamically-defined XML element's attribute.
    - Jsp:body - Defines dynamically-defined XML element's body.
    - jsp:text

- How JSP works? (8 Steps)
- JSP Life-cycle
    - What happens at page translation time? (Show Practically with scripting and Directive)

        JSP constructs get translated into servlet code

        Page translation does not occur for each request

- Hidden / HTML Comment
    - <!-- comment [<%= expression%>] -->
    - <%-- comment→

J2EElecture- Servlet

- Overview of Web application, Servlet technology
    - 3-Tier Architectures
    - Why JSP
    - Benefits of JSP
- Writing your first servlet
    - Servlet LifeCycle
        - *Loading Servlet*
            - *Loading : Loads the Servlet class.*
            - *Instantiation : Creates an instance of the Servlet*
        - init() - Servlet container initializes the instantiated Servlet object
            - Servlet.init(**ServletConfig**)
            - This method is used to initialize the resources, such as JDBC datasource.
        - service()
            - Servlet.service(**ServletRequest**, **ServletResponse**)
        - destroy()
            - destroy()
- Running and debugging Servlets
- Handling the client request
- Form data, retrieve parameters
    - getParameter("name")
    - getParameterValues("name")
    - getParameterNames();

- Understand HTTP, HTTP request headers (**Practical Examples**)
    - Request Header
        - Accept,
        - Accept-Encoding
        - Connection, Referrer, User-Agent
        - https://developer.mozilla.org/en-US/docs/Glossary/Request_header
- Generating the server response
    - response.setStatus(404)
    - `response.setContentType("text/html");`
    - `response.setIntHeader("Refresh", 5);`
    - response.setHeader("Cache-Control", "no-cache");
    - etc..
- HTTP status codes
    - 200 - OK
    - 404 - NOT FOUND
    - 500 - SERVER ERROR
- HTTP response headers
    - https://developer.mozilla.org/en-US/docs/Glossary/Response_header
- Advanced Servlet Concepts
- HTTP Redirects
    - response.sendRedirect("path");
    - request.getRequestDispatcher("welcome.jsp").forward(request, response);
    - response.sendError(0);

    - response.sendError(0,"msg");

Why Cookies?

- Handling cookies
    - Cookie cookie = new Cookie("myCookie", "myCookieValue");
    - response.addCookie(cookie);
    - ---
    - .
    - Cookie[] cookies = request.getCookies();
    - if (cookies != null) {
    -   for(int i=0; i<cookies.length; i++) {
    -     Cookie c = cookies[i];
    -     if (c.getName().equals("someName")) {
    -       doSomethingWith(c);
    -       break;
    -     }
    -   }
    - }
    - .
    - c.getName(); //Get Name
    - c.getValue(); //Get Value
    - cookie.setMaxAge(3600); //Set Time

- Session tracking
    - HttpSession session = **request.getSession(true);**
    - ShoppingCart cart = (ShoppingCart)session.**getAttribute**("shoppingCart");
    - session.**setAttribute**("shoppingCart", cart);

      More..

    - isNew
    - getCreationTime
    - getLastAccessedTime
    - getMaxInactiveInterval,
    - setMaxInactiveInterval
    - **invalidate**