Request header example:

GET / HTTP/1.1

Host: www.facebook.com

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.3

Connection: keep-alive

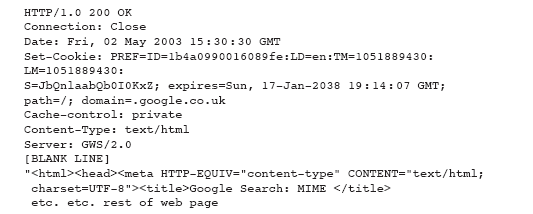
Cookie: sb=xyz123; datr=abc456; fr=0a1b2c3d4e5f6g7h8i9j

Referer: https://www.google.com/

If-Modified-Since: Wed, 21 Oct 2020 07:28:00 GMT

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36

* It can also send a number of headers:
  + Accept The MIME types the browser prefers.
  + Accept-Charset The character set the browser expects.
  + Content-Length (for POST messages, how much data is attached)
  + Connection Use persistent connection? If a servlet gets a Keep-Alive.
  + Cookie (one of the most important headers)
  + Host (host and port as listed in the original URL)
  + If-Modified-Since (only return documents newer than this)
  + Referer (the URL of the page containing the link the user followed to get to current page)
* HTTP 1.1 defines the following request methods(7):
  + **GET** - retrieves the resource identified by the request URL.
  + **HEAD** - returns the headers identified by the request URL.
  + **POST** - sends data of unlimited length to the web server.
  + **PUT** - stores a resource under the request URL.
  + **DELETE** - removes the resource identified by the request URL.
  + **OPTIONS** - returns the HTTP methods the server supports.
  + **TRACE** - returns the header fields sent with the TRACE request.

 **RESPONSE**

Some commonly used status codes include:

* + **404 *- indicates that the requested resource is not available.***
  + **401 *- indicates that the request requires HTTP authentication.***
  + **500 *- indicates an error inside the HTTP server which prevented it from fulfilling the request.***
  + **503 *- indicates that the HTTP server is temporarily overloaded, and unable to handle the request.***

1. *Switching Protocols*

*Server will comply with Upgrade header and change to different protocol. (New in HTTP 1.1)*

1. *OK*

*Everything's fine; document follows for GET and POST requests. This is the default for servlets; if you don't use setStatus, you'll get this.*

1. *Created*

*Server created a document; the Location header indicates its URL.*

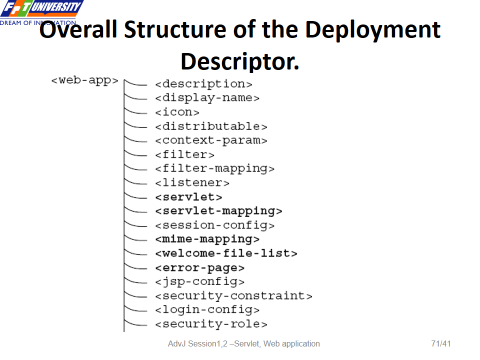
202 *Accepted*

*Request is being acted upon, but processing is not completed.*

203 *Non-Authoritative Information*

*Document is being returned normally, but some of the response headers might be incorrect since a document copy is being used.*

***Construct the file and directory structure of a Web Application that may contain:***

* + *static content,*
  + *JSP pages,*
  + *servlet classes,*
  + *the deployment descriptor (web.xml)*
  + *tag libraries,*
  + *JAR files and Java class files;*
  + 

**II, JSP**

**How JSP work:**

1. **Request**: User requests a JSP page.
2. **Translation**: JSP file is translated into a Java servlet source file. (.java)
3. **Compilation**: Java servlet source file is compiled into a servlet class. (.class)
4. **Loading**: Compiled servlet class is loaded into memory.
5. **Execution**: Servlet's service method generates and sends the response.
6. **Subsequent Requests**: For subsequent requests, the already compiled servlet class (.class file) is used unless the JSP file is modified. If modified, the process from translation to execution is repeated.

**JSP Directives (Directives, which—like scripting elements—are pieces of JSP tag-like syntax.):**

1. Page <%@ page attributeName=“value" %>
2. Import <%@ include file=“**relativeURL** " %>
3. taglib (JSP tag-like syntax)

* **JSP Scripting Elements:** *declaration* <%! %> (declare a variable or method) , *expression* <%= %> ( the value is converted to string ), *scriptlet* <% %> (When this JSP code is translated into a servlet, it will be placed inside the \_jspService(HttpServletRequest request, HttpServletResponse response) method )

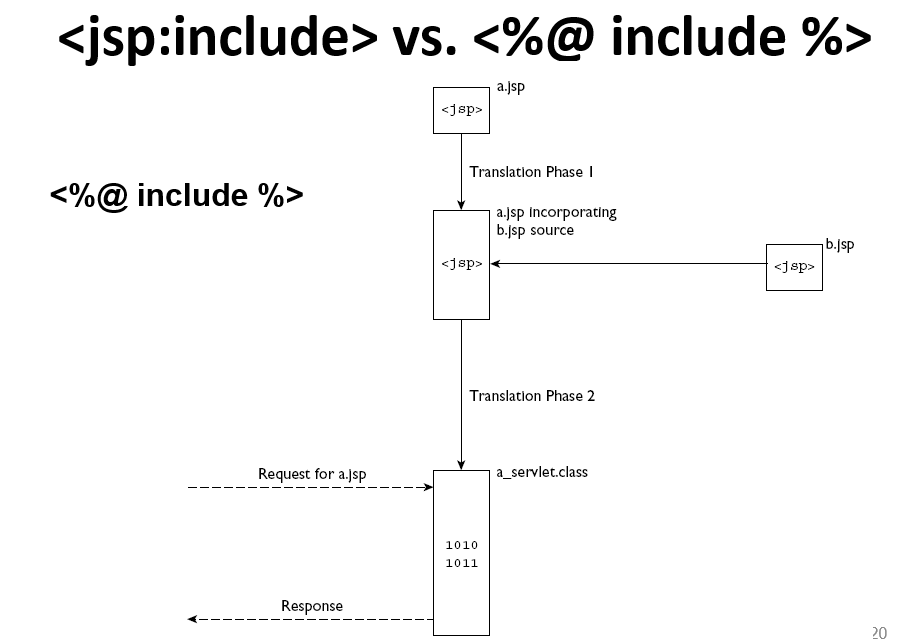
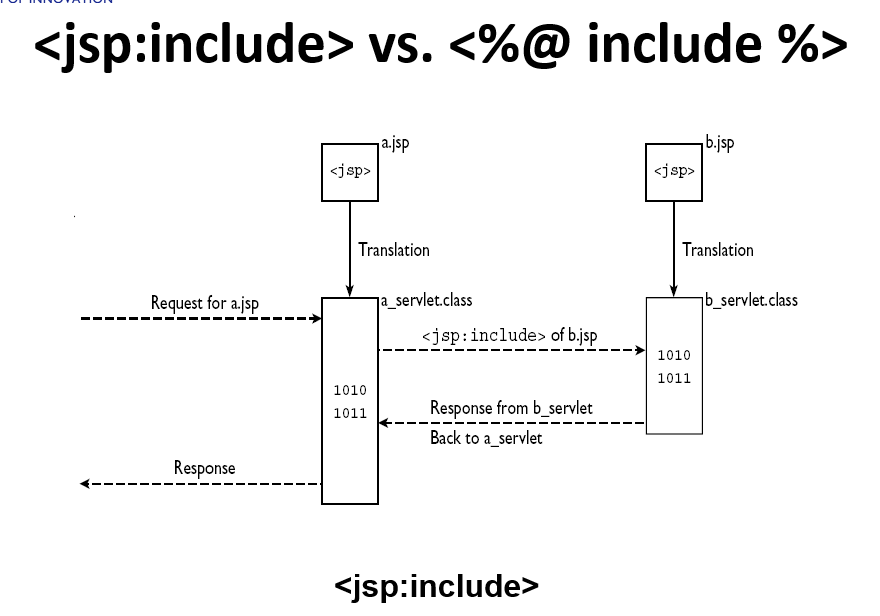
**JSP predefined-variable:**

* *request –* Object of HttpServletRequest (request parameters, HTTP headers, cookies
* *response –* Object of HttpServletResponse
* *out -* Object of PrintWriter buffered version JspWriter
* *session -* Object of HttpSession associated with the request
* *application -* Object of ServletContext shared by all servlets in the engine
* *config -* Object of ServletConfig
* *pageContext -* quản lý thông tin và trạng thái của trang JSP hiện tại.
* *page –* variable synonym for this object

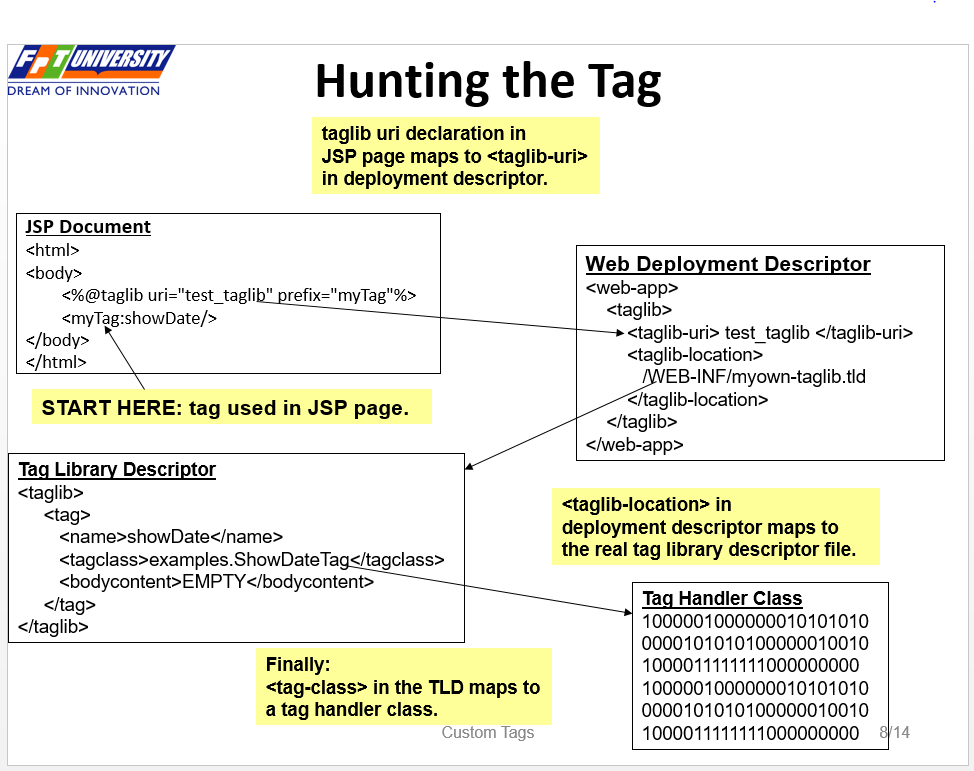
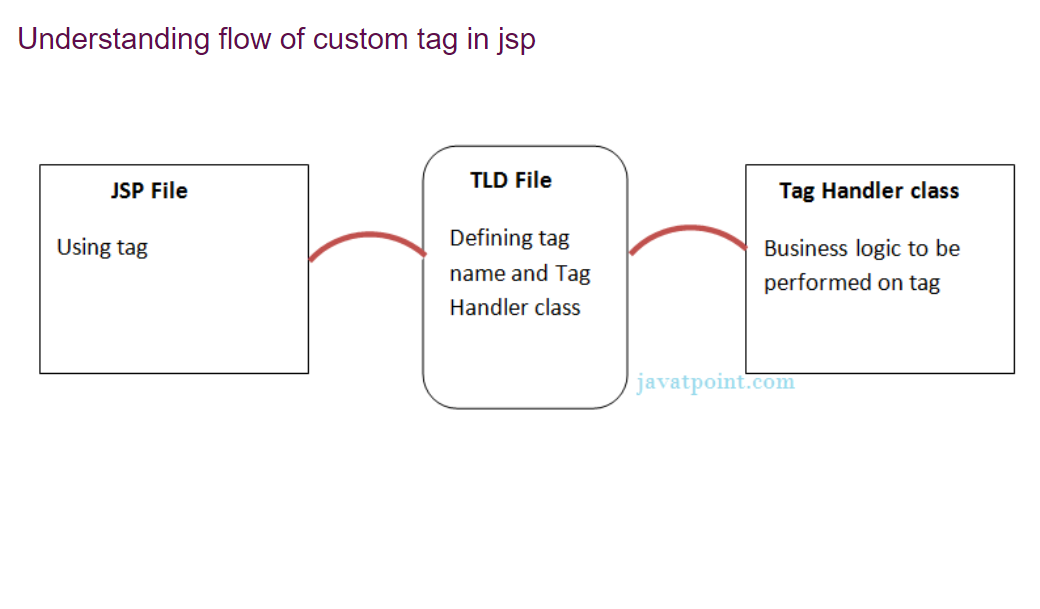
**Ngoài ra jsp còn có standard actions ( thay thế scripting element để thực thi công việc 1 cách gọn gàng):**

|  |  |
| --- | --- |
| **JSP Action Tags** | **Description** |
| jsp:forward | forwards the request and response to another resource. |
| jsp:include | includes another resource. |
| jsp:useBean | creates or locates bean object. |
| jsp:setProperty | sets the value of property in bean object. |
| jsp:getProperty | prints the value of property of the bean. |
| jsp:plugin | embeds another components such as applet. |
| jsp:param | sets the parameter value. It is used in forward and include mostly. |
| jsp:fallback | can be used to print the message if plugin is working. It is used in jsp:plugin. |

<jsp:include page="includedPage.jsp" /> vs <%@ include file=“ includedPage.jsp " %> (include directive) khác nhau như sau:



<jsp:include> is evaluated at **request time**, not at translation time.



JSTL ~ jsp standard action ~ custom tag , EL , Directive , scripting elements (are 6 syntax types in jsp)

**JavaBean** là các class Java thuần, dùng để thể hiện các thực thể trong các chương trình Java.

Một lớp JavaBean sẽ có 3 đặc điểm như sau về mặt cú pháp:

*1. Là một public class  
2. Có các thuộc tính private  
3. Ứng với mỗi thuộc tính, lớp sẽ cung cấp một cặp phương thức setter / getter để truy cập và thao tác với giá trị của từng thuộc tính.*

