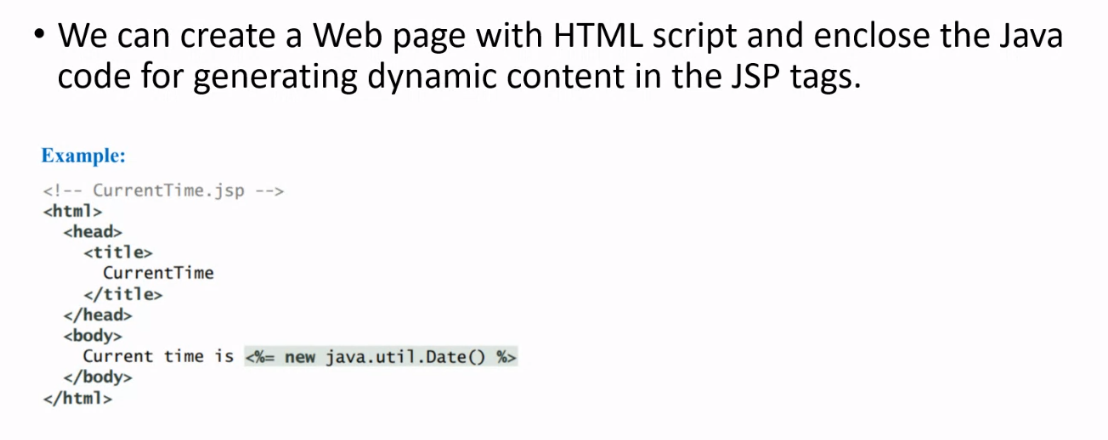
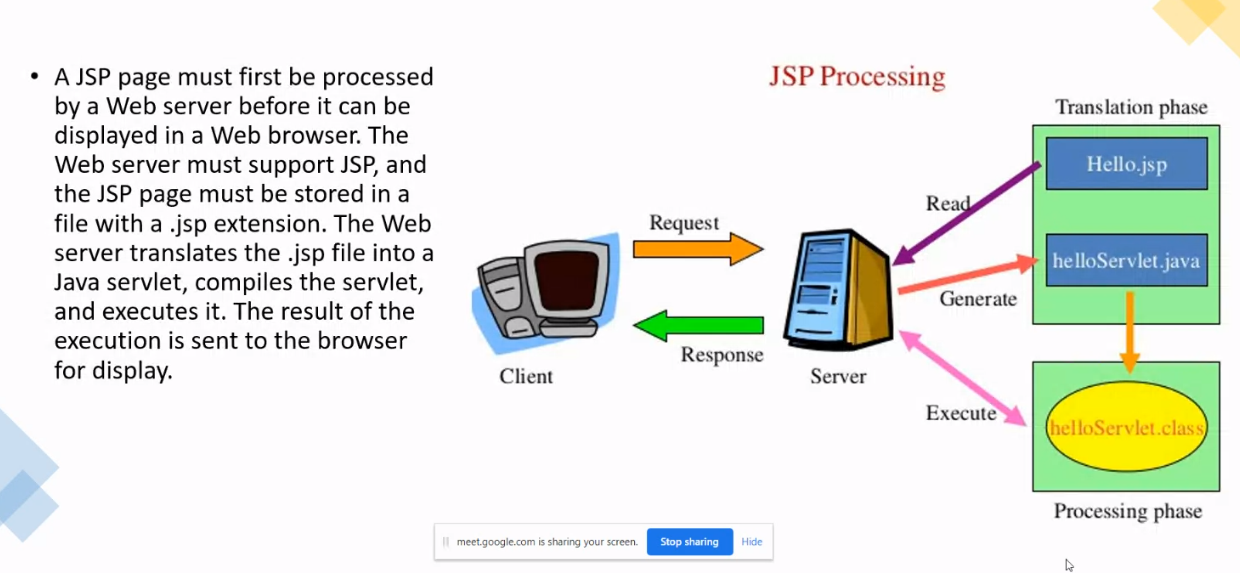
**Tutorial 5**

1. With example, describe how you can ***create*** a simple ***JSP page***
   1. Create a new Web Project
   2. Right-click the Web project node in the project pane and choose New > JSP to display the New JSP dialog box.
   3. Enter your *<JSP file name>* in the JSP File Name field and click Finish. You will see *<JSP file name>*.jsp appearing under the Web Pages node in the Web project.
   4. Complete the code for *<JSP file name>.jsp*.
   5. Right-click *<JSP file name.jsp>* in the project pane and choose Run File. The JSP page will be displayed in a Web browser

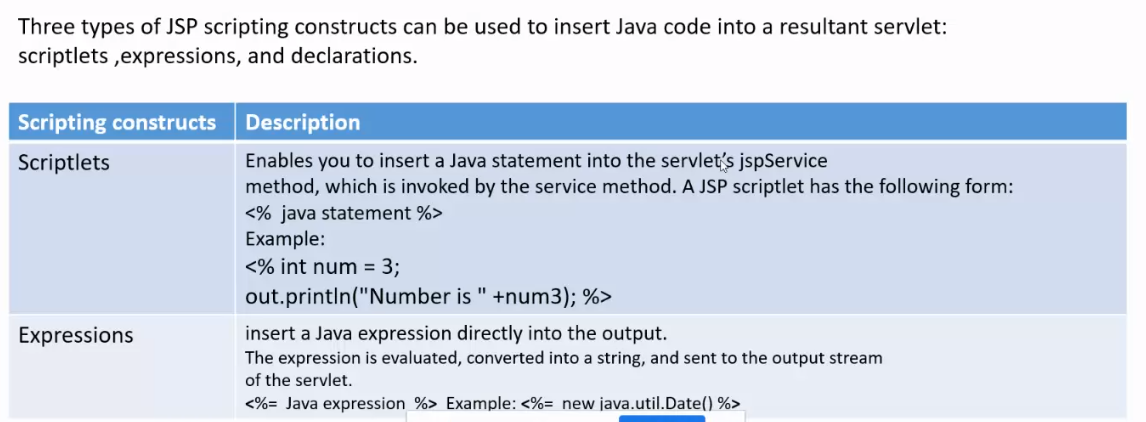


1. Explain ***how*** a ***JSP page***is ***processed***.
2. Translate the .jsp file into a Java servlet.
3. Compile the servlet.
4. Execute the servlet. The result of the execution is sent to the browser for display



3. List and describe the use **JSP scripting** constructs can be used to insert Java code into a resultant servlet.

* **JSP Expression**
* Used to insert a Java expression directly into the output
* Format: *<%= Java-expression %>*
* **JSP Scriptlet**
* Used to insert a Java statement into the servlet’s jspService method which invoked by service method
* Format: *<% Java statement %>*
* **JSP Declaration**
* Used for declaring methods or fields into the servlet
* Format: *<%! Java Method or field declaration %>*

**

1. JSP provides eight predefined variables from the servlet environment that can be used with JSP expressions and scriptlets. specify THREE (3) use of predefined variables.

**request**

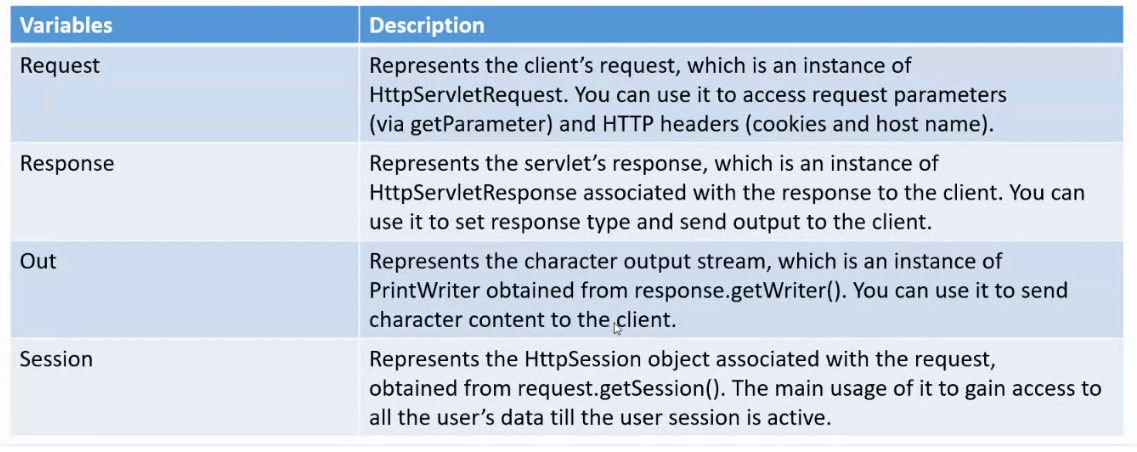
* Represents the client’s request, which is an instance of HttpServletRequest. You can use it to access request parameters, HTTP headers such as cookies, hostname, etc.

**response**

* Represents the servlet’s response, which is an instance of HttpServletResponse. You can use it to set a response type and send output to the client.

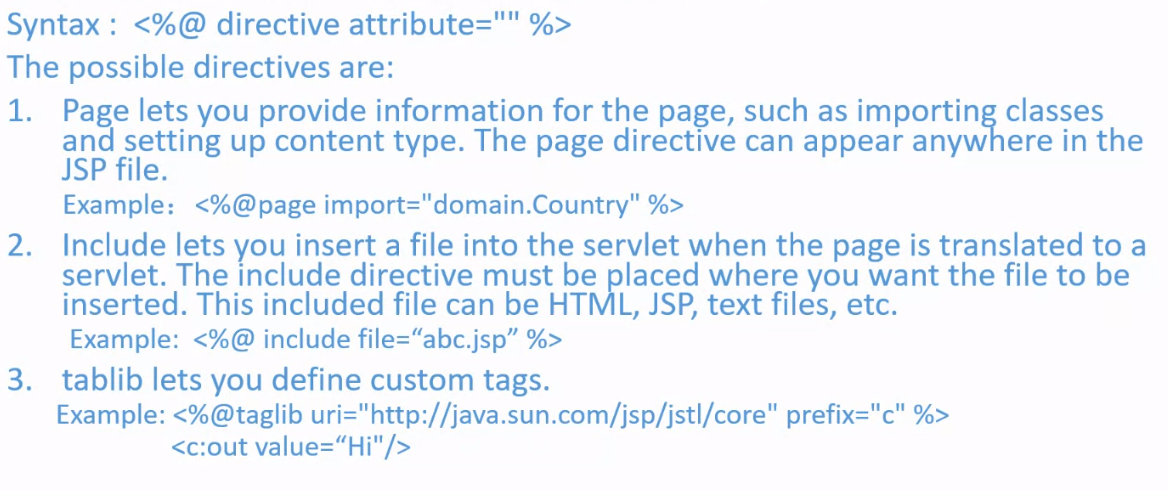
**out**

* Represents the character output stream, which is an instance of PrintWriter obtained from response.getWriter() . You can use it to send character content to the client.



1. A **JSP directive** is a statement that gives the JSP engine information about the JSP page. Describe the use of directives in JSP.

* **page**
* Lets you provide information for the page, such as importing classes and setting up content type. The page directive can appear anywhere in the JSP file.
* **include**
* Lets you insert a file to the servlet when the page is translated to a servlet. The include directive must be placed where you want the file to be inserted.
* **taglib**
* Let you define custom tags



1. Write syntax to create an **instance** for a **JavaBeans component**.

| <jsp:usebean id= “objectName” scope= “scopeAttribute” class= “ClassName” /> |
| --- |

1. A **JavaBeans component** provides the **get and set methods** for reading and modifying its private properties. Show how you can get and set the property in JSP using syntax.

Get Property in JSP:

| <jsp:getProperty name= “beanId” property= “age” /> |
| --- |

Set Property in JSP:

| <jsp:setProperty name= “beanId” property= “age” value= “30” /> |
| --- |

1. Suppose you want to *get the value of the input parameter* named score and *set it to the JavaBeans property* named score. Write the code to show the properties are associated with input parameters.

| <% double score =  Double.parseDouble(request.getParameter(“score”))  />  <jsp:setProperty name= “beanId” property= “score” value= “<%= score %>”/> |
| --- |

1. Web applications developed using JSP generally consist of many pages linked together. Show how you can **forward a page to another page** by using syntax.

| <jsp:forward page= “destination” /> |
| --- |