OBJECT™ TECHNOLOGIES

JSP II

What will be covered

OBJECT

- Implicit objects in JSP
- Scripting elements
- Drawback of scripting elements
- Importance of scriptless JSP elements
- Details of JSP directives
- Page directive and it's attributes

Implicit objects in JSP

OBJECT

- These objects are created by JSP Engine during translation phase (while translating JSP to Servlet).
- They are being created inside service method so we can directly use them within Scriptlet without initializing and declaring them.
- There are total 9 implicit objects available in JSP.

Implicit object	Type	Purpose
out	javax.servlet.jsp.JspWriter	used for writing content to the client
request	javax.servlet.http.HttpServlet Request	purpose of request implicit object is to get the data on a JSP page which has been entered by user on the previous JSP page
response	javax.servlet.http.HttpServlet Response	used for modfying or delaing with the response which is being sent to the client(browser) after processing the request.

Implicit objects in JSP

OBJECT

Implicit object	Type	Purpose
session	javax.servlet.http.HttpSession	used for storing the user's data to make it available on other JSP pages till the user session is active.
application	javax.servlet.ServletContext	used for getting application-wide initialization parameters and to maintain useful data across whole JSP application.
exception	javax.servlet.jsp.JspException	used in exception handling for displaying the error messages. This object is only available to the JSP pages, which has isErrorPage set to true.
page	java.lang.Object	is a reference to the current Servlet instance (Converted Servlet, generated during translation phase from a JSP page)

Scripting elements

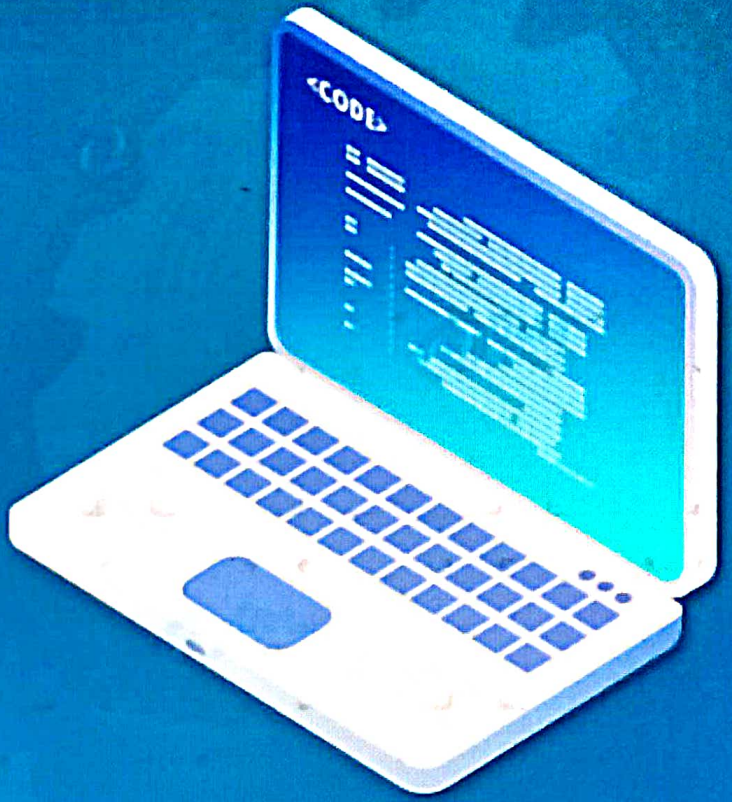
OBJECT

JSP Scripting Elements

01 *Expressions*

02 *Scriptlets*

03 *Declarations*



Scripting elements

OBJECT

- JSP scripting elements are one of the most vital elements of a JSP code.
- <% %> tags are of the utmost importance as, at the time of translation, the JSP engine will consider anything inside these tags. Only this code will convert to Java code. Code other than this is plain or HTML text.
- The scripting elements thus help to embed java code to the HTML, CSS, JavaScript code.
- Comments increase the understanding of the code. They will not be there in the output. Browser or the container ignores them.
- The jsp directives help the web container to translate a JSP page into the corresponding servlet. e.g. specifying what classes to import etc.

Scripting elements

```
<%
    SimpleDateFormat sdf = new SimpleDateFormat("HH");
     int hh =
Integer.parseInt(sdf.format(Calendar.getInstance().getTime()));
     if(hh < 12 )
     {   %>
          <%= "<h1> Good Morning </h1>" %>
     <%}
     else if(hh >= 12 && hh <= 16)
     { %>
          <%=  "<h1> Good Afternoon </h1>" %>
     <%}
     else if(hh > 16 && hh <= 20)
     { %>
          <%=  "<h1> Good Evening </h1>" %>
     <% }
     else
     { %>
          <%=  "<h1> Good Night </h1>"  %>
     <% }
     %>
```

Drawback of scripting elements

OBJECT

- JSP Scriptlets reduces the maintainability and readability of the code and hence making it difficult to read and make further changes.
- JSP Scriptlets can not be re-used.
- JSP Scriptlets are not unit-testable.
- JSP Scriplets merge the presentation with business logic which is highly prone to errors and thus break the basic principle of MVC model architecture.
- If JSP Scriptlets throws an exception, it breaks the whole page there and doesn't move further in the processing.
- Nesting of scripting elements into one another is not possible which makes code very clumpsy to read.

Importance of scriptless JSP elements

OBJECT

- The primary purpose of a JSP is to serve as a template for rendering the presentation format (usually HTML) as the results of a request.
- In many cases, decisions need to be made, using the submitted data and other available information.
- With the emergence of a finalized JSP Specification, basic bean-manipulation standard actions were defined that still exist to this day.
- Moving the processing code out of the pages and into Java classes was a good way of eliminating some of the problems introduced by having on-page code.
- Scriptless JSP can be effective to keep scripting elements away from JSP pages.

Importance of scriptless JSP elements

OBJECT

- If a clear separation between your presentation and business logic is expected, then force the page to go scriptless. By enforcing scriptless pages, the dynamic behavior of JSP pages must be provided through other elements such as JavaBeans, EL Expressions, Custom actions and standard tag libraries.
- **JSP standard actions**
- JSP standard actions are powerful. They can avoid lot of boilerplate scriptlet code and ease JSP code development. JSP standard actions are great time-savers and help in JSP view generation.
- **EL**
- EL provides the ability to use run-time expressions outside JSP scripting elements. An Expression Language makes it possible to easily access application data stored in JavaBeans components without any scriptlet code.

Importance of scriptless JSP elements

OBJECT

- **JSTL**
- The JavaServer Pages Standard Tag Library (JSTL) is a collection of useful JSP tags which encapsulates the core functionality common to many JSP applications.
- JSTL has support for common, structural tasks such as iteration and conditionals, tags for manipulating XML documents, internationalization tags, and SQL tags.
- **Custom tags**
- They are user-defined tags. They eliminates the possibility of scriptlet tag and separates the business logic from the JSP page.
- The same business logic can be used many times by the use of custom tag.

Details of JSP directives

OBJECT

- JSP directives are the messages to JSP container. They provide global information about an entire JSP page.
- JSP directives are used to give special instruction to a container for translation of JSP to servlet code.
- In JSP life cycle phase, JSP has to be converted to a servlet which is the translation phase.
- They give instructions to the container on how to handle certain aspects of JSP processing
- Directives can have many attributes by space separated as key-value pairs.
- In JSP, directive is described in <%@ %> tags.
- There are three types of directives:
  - Page directive
  - Include directive
  - Taglib directive

Details of JSP directives

OBJECT

- **Template text**

  JSP is complementary for servlet

- Text without any markups is considered as template text. It is translated similarly like HTML tags

- **Scriptlets**

```
<%
    out.println("Your IP address is " + request.getRemoteAddr());
%>
```

- A scriptlet can contain any number of JAVA language statements, variable or method declarations, or expressions that are valid in the page scripting language.

- At the time of translation phase, only symbols are removed and content of scriptlet are added in service method of generated servlet

JSP elements

OBJECT

- Basically, Page directives are used to give the instructions to the JSP compiler like what package to import, what language we are writing, etc. Page directives are basically used for supplying compile-time information to the container for generating a servlet. Page directive should be first statement in JSP. The scope of the page directive is applicable to the current JSP page only. JSP program allows only one-page directive at a time.
- The include directive is employed to incorporate a file during the interpretation phase. This directive tells the container to merge the content of other external files with the present JSP. This tag is given for code inclusion, not for output inclusion.
- The main purpose of Taglib Directives is to make available user-defined tag library into the present JSP pages.

JSP elements

OBJECT

- **Expressions**
- <%= expression %>
- A JSP expression element contains a scripting language expression that is evaluated, converted to a String, and inserted where the expression appears in the JSP file.
- The expression element can contain any expression that is valid according to the Java Language Specification but you cannot use a semicolon to end an expression.
- At the time of compilation, expressions get converted to out.print() statement and added in the service method of generated servlet.
- Today's date: <%= (new java.util.Date()).toLocaleString()%>
- **JSP Declarations**
- A declaration declares one or more variables or methods that you can use in Java code later in the JSP file. You must declare the variable or method before you use it in the JSP file

JSP elements

OBJECT

- <%! declaration; [declaration;]+ ... %>.
- The jsp declaration tag can declare variables as well as methods.
- The declaration of jsp declaration tag is placed outside the _jspService() method in the generated servlet class.
- **JSP Directives**
- A JSP directive affects the overall structure of the servlet class. It usually has the following form –
- <%@ directive attribute="value" %>
- There are three types of directive tag – page, include, taglib.
- Directives can have a number of attributes which can be listed down as key-value pairs and separated by spaces.
- **Std Actions**
- **EL**
- **JSTL**
- **Custom tags**

JSP II

Now as we know how JSP complements servlets, let us understand more details about JSP technology. How different elements can be used together to get response generated.

Using Scriptlet, Expression and Declaration

```
<%@page import="java.util.Calendar,java.text.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"  pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
  <%
     SimpleDateFormat sdf = new SimpleDateFormat("HH");
     String hours = sdf.format(Calendar.getInstance().getTime());
     int hh = Integer.parseInt(hours);
     if(hh < 12)
     {  %>
        <%= "<h1> GOOD MORNING </h1>" %>
    <% }
     else if(hh >= 12 && hh <= 16)
     { %>
        <%= "<h1> GOOD AFTERNOON </h1>" %>
    <% }
     else if(hh > 16 && hh <= 21)
     { %>
        <%= "<h1> GOOD EVENING </h1>" %>
    <%}
     else
     { %>
        <%= "<h1> GOOD NIGHT </h1>" %>
    <% }
  %>
</body>
```

Using Scriptlet, Expression and Declaration

```
<%@page import="java.util.Calendar,java.text.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"  pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
  <%
     SimpleDateFormat sdf = new SimpleDateFormat("HH");
     String hours = sdf.format(Calendar.getInstance().getTime());
     int hh = Integer.parseInt(hours);
     if(hh < 12)
     {  %>
        <%= "<h1> GOOD MORNING </h1>" %>
     <% }
     else if(hh >= 12 && hh <= 16)
     { %>
        <%= "<h1> GOOD AFTERNOON </h1>" %>
     <% }
     else if(hh > 16 && hh <= 21)
     { %>
        <%= "<h1> GOOD EVENING </h1>" %>
     <%}
     else
     { %>
        <%= "<h1> GOOD NIGHT </h1>" %>
     <% }
  %>
</body>
</html>
```

Above JSP greets the user on the basis of current time. It makes use of scriptlets for java code and expression for generating HTML. Major disadvantage is scriptlet, expression and declaration can not be embedded into one another. Hence the readability of JSP code gets reduced which affects maintainance.

Above JSP greets the user on the basis of current time. It makes use of scriptlets for java code and expression for generating HTML. Major disadvantage is scriptlet, expression and declaration can not be embedded into one another. Hence the readability of JSP code gets reduced which affects maintainance.

So scriptless JSP elements are preferred in JSP like JSP actions, EL and custom tags.

JSP Directives

JSP directives are instructions provided to the JSP engine about various aspects of the page. They are placed inside <%@ ... %> tags and play a crucial role in configuring and managing the behavior of your JSP pages.

There are three types of directives:

1. page directive
2. include directive
3. taglib directive

Syntax of JSP Directive

```
<%@ directive attribute="value" %>
```

The page directive

It is one of the most important JSP directives, offering fine-grained control over various aspects of your JSP page. Different attributes that can be used with this directive are :

1. Language: Specifies the scripting language used in the page (e.g., Java, JavaScript).
2. contentType: Defines the content type of the response sent by the page (e.g., text/html, text/plain).
3. errorPage: Sets the page to redirect to in case of errors during page execution.
4. import: Lists Java classes or packages to be imported throughout the page, providing direct access within scriptlets and expressions.
5. buffer: Configures the buffering behavior for the response.
6. isELIgnored: Specifies whether Expression Language should be evaluated or ignored within the page.
7. session: Controls whether the page participates in the current HTTP session.
8. threadSafe: Defines whether the generated servlet for the page can handle concurrent requests safely.

Commonly used attributes:

These attributes are set with some default values automatically. If required this value can be changed as per the requirements.

Attribute	Value

The page directive

It is one of the most important JSP directives, offering fine-grained control over various aspects of your JSP page. Different attributes that can be used with this directive are :

1. Language: Specifies the scripting language used in the page (e.g., Java, JavaScript).
2. contentType: Defines the content type of the response sent by the page (e.g., text/html, text/plain).
3. errorPage: Sets the page to redirect to in case of errors during page execution.
4. import: Lists Java classes or packages to be imported throughout the page, providing direct access within scriptlets and expressions.
5. buffer: Configures the buffering behavior for the response.
6. isELIgnored: Specifies whether Expression Language should be evaluated or ignored within the page.
7. session: Controls whether the page participates in the current HTTP session.
8. threadSafe: Defines whether the generated servlet for the page can handle concurrent requests safely.

Commonly used attributes:

These attributes are set with some default values automatically. If required this value can be changed as per the requirements.

Attribute	Value
language	java
contentType	text/html; charset=ISO-8859-1
buffer	8kb (8 kilobytes buffer size)
isELIgnored	false (Expression Language is enabled by default)
isThreadSafe	true (generated servlet is thread-safe by default)
session	true (page participates in the current HTTP session)
isErrorPage	false (page is not an error page by default)

Note : The page directive is processed during the translation phase, influencing the servlet generated for the JSP page.

The include directive

The include directive allows you to incorporate content from another file into a JSP page during translation, creating a single, unified response. This included file can be HTML, JSP, text files, etc.

The include directive

The include directive allows you to incorporate content from another file into a JSP page during translation, creating a single, unified response. This included file can be HTML, JSP, text files, etc.

It's useful for:

- Reusing common elements (headers, footers, navigation menus) across multiple pages.
- Modularizing code for better maintainability.
- Conditionally including content based on runtime conditions.

Syntax is :

```
<%@ include file="relativeURL" %>
```

Note :

- During translation, the JSP engine replaces the directive with the contents of the included file.
- The included content becomes part of the current JSP page as if it were written directly within it.
- Variables and objects declared in the including page are accessible within the included file.

The taglib directive

It allows you to incorporate custom tags from external tag libraries into your JSP pages, providing reusable components and extending functionality. It acts as a bridge between your JSP page and the tag library, enabling the use of custom tags defined within the library.

Syntax:

```
<%@ taglib uri="uri" prefix="prefix" %>
```

uri: Specifies the unique URI (usually a URL) that identifies the tag library.

prefix: Assigns a short prefix to reference the tags from the library within the page.

Steps to use a taglib:

1. Identify a tag library: Choose a library that suits your needs (e.g., JSTL, custom libraries).
2. Include the taglib directive: Add it to the top of your JSP page, providing the URI and prefix.
3. Use the tags: Employ the tags from the library within your page, prefixed with the chosen prefix.

Importance Of Scriptless JSP Elements

Scriptless JSP elements refer to elements that allow dynamic content

1. Identify a tag library: Choose a library that suits your needs (e.g., JSTL, custom libraries).
2. Include the taglib directive: Add it to the top of your JSP page, providing the URI and prefix.
3. Use the tags: Employ the tags from the library within your page, prefixed with the chosen prefix.

Importance Of Scriptless JSP Elements

Scriptless JSP elements refer to elements that allow dynamic content generation without directly embedding Java code within the page. They promote a cleaner separation of concerns, better maintainability, and enhanced security. Following is the limitations of scriptlets :

- Scriptlets (<% %>) mix Java code with HTML, making pages harder to read and maintain.
- Java code within scriptlets can become complex and difficult to manage, especially for non-Java developers. Scriptless elements often provide simpler syntax and abstractions for common tasks (e.g., conditional logic, iteration, formatting).
- Scriptlets can introduce vulnerabilities if not properly validated and sanitized.
- Pages with excessive scriptlets can become cluttered and difficult to follow.

Scriptless JSP elements is generally considered a best practice:

- It leads to cleaner, more maintainable, and secure code.
- It promotes separation of concerns and better collaboration between designers and developers.
- It encourages the use of reusable components and best practices.

Following are the scriptless JSP elements :

1. Std(JSP) actions
2. Expression Language(EL)
3. JSTL(JSP Standard Tag Library)
4. Cutom tags

Assignments

1. Greet the user by using right message based on time like 'Good Morning'.
2. Replace the servlet in shopping application which displays the list of cart items to the user with jsp(use scriptlet, expression and declaration).
3. Replace the servlet in shopping application which allows user to select the product and add it in the cart by jsp (use scriptlet, expression and declaration).
4. Use error page for JSP for handling exceptions generated by JSP. Use exception object for error page