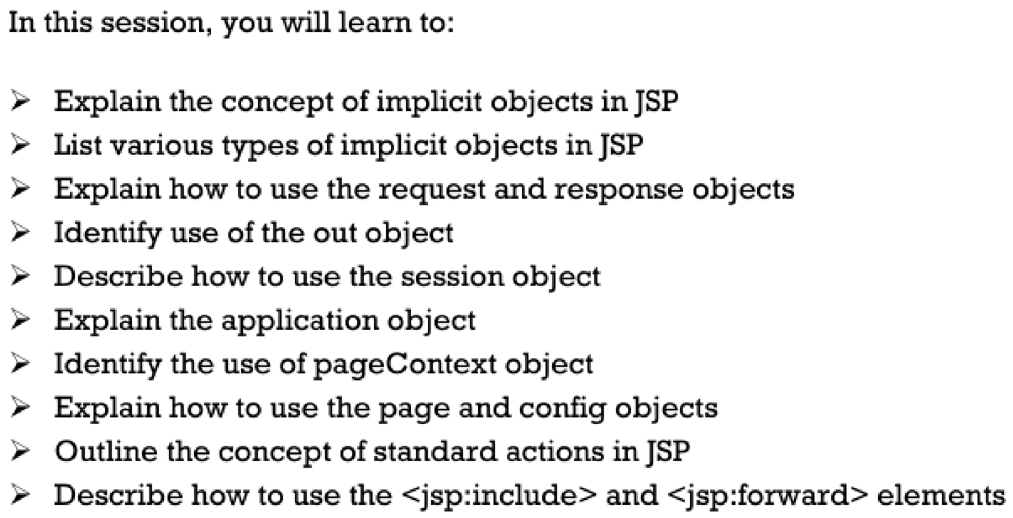
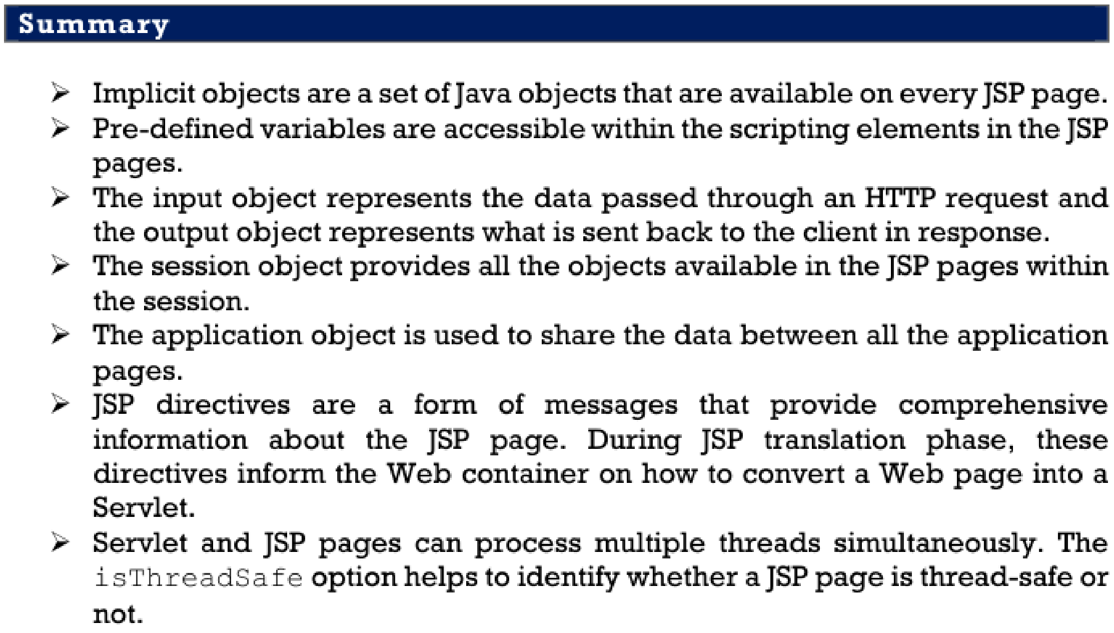
JakEE JSP Implicit Objects and Standard Actions



In this session, you will learn to:

- Explain the concept of implicit objects in JSP
- List various types of implicit objects in JSP
- Explain how to use the request and response objects
- Identify use of the out object
- Describe how to use the session object
- Explain the application object
- Identify the use of pageContext object
- Explain how to use the page and config objects
- Outline the concept of standard actions in JSP
- Describe how to use the <jsp:include> and <jsp:forward> elements

image-126.png



Summary

- Implicit objects are a set of Java objects that are available on every JSP page.
- Pre-defined variables are accessible within the scripting elements in the JSP pages.
- The input object represents the data passed through an HTTP request and the output object represents what is sent back to the client in response.
- The session object provides all the objects available in the JSP pages within the session.
- The application object is used to share the data between all the application pages.
- JSP directives are a form of messages that provide comprehensive information about the JSP page. During JSP translation phase, these directives inform the Web container on how to convert a Web page into a Servlet.
- Servlet and JSP pages can process multiple threads simultaneously. The `isThreadSafe` option helps to identify whether a JSP page is thread-safe or not.

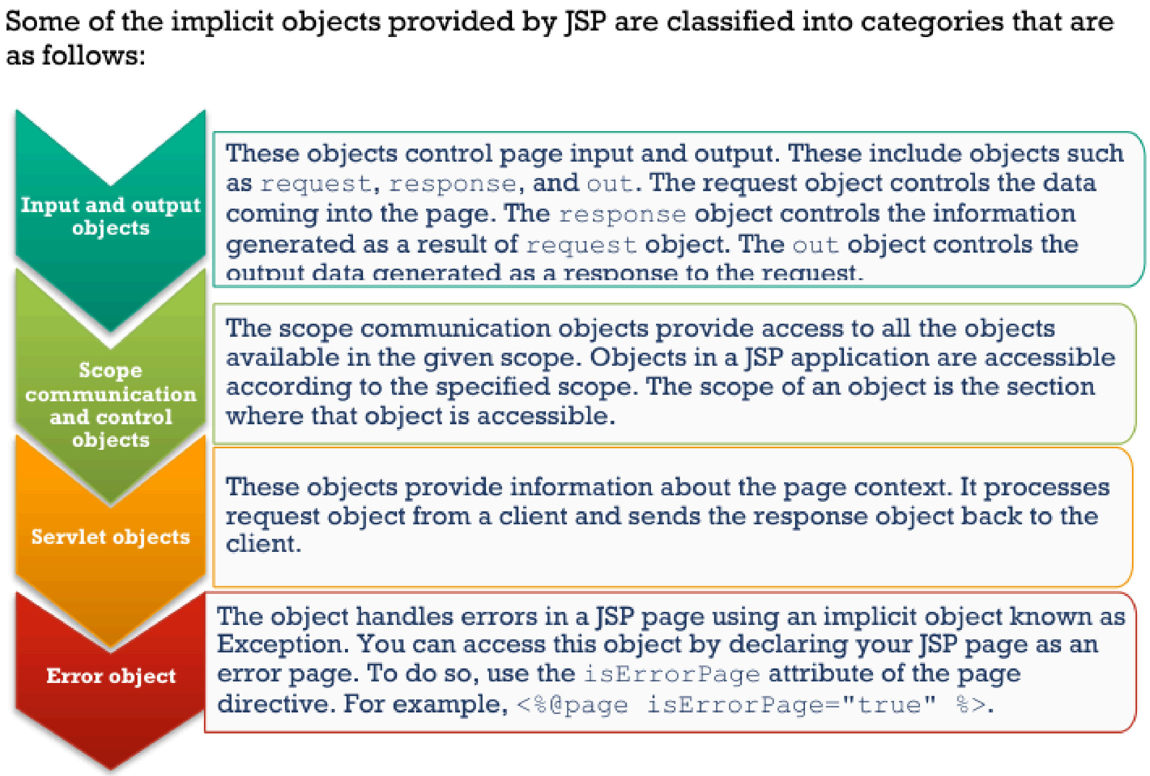
image-129.png

Notables

- When the JSP page is translated into a Servlet class, all the implicit objects declarations are taken within the `_jspService() method`
- Implicit objects work like setting a middleware function at the top of the execution stack to preset or make certain attributes

available throughout the JSP

Types of Implicit Objects



Some of the implicit objects provided by JSP are classified into categories that are as follows:

Input and output objects

These objects control page input and output. These include objects such as request, response, and out. The request object controls the data coming into the page. The response object controls the information generated as a result of request object. The out object controls the output data generated as a response to the request.

Scope communication and control objects

The scope communication objects provide access to all the objects available in the given scope. Objects in a JSP application are accessible according to the specified scope. The scope of an object is the section where that object is accessible.

Servlet objects

These objects provide information about the page context. It processes request object from a client and sends the response object back to the client.

Error object

The object handles errors in a JSP page using an implicit object known as Exception. You can access this object by declaring your JSP page as an error page. To do so, use the isErrorPage attribute of the page directive. For example, `<%@page isErrorPage="true" %>`.

image-127.png

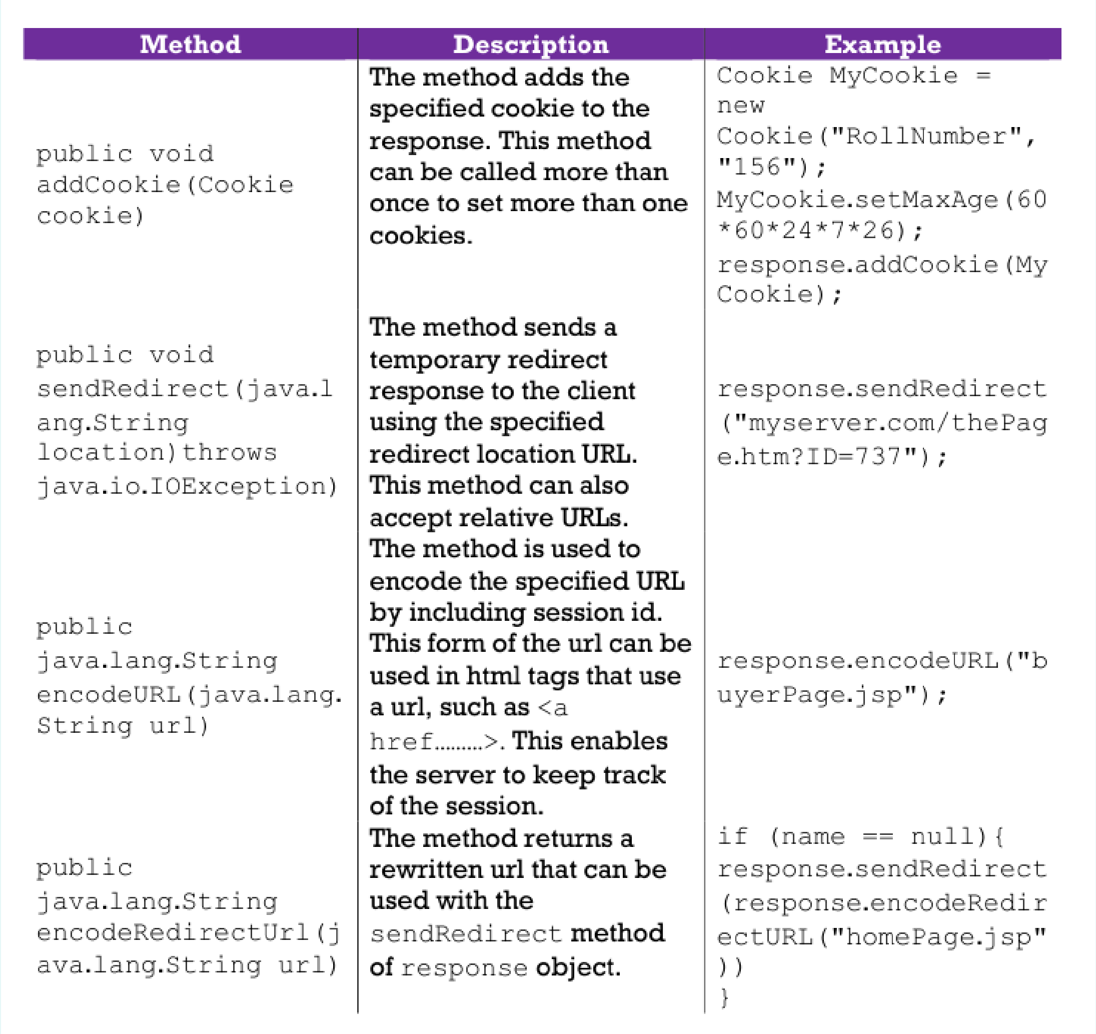
Input and Output Objects

notables

Methods in Req object

- `getRemoteHost()` - gets the host name
- `getRemoteAddress()` - gets the IP address

Methods in Res object

- 

Method	Description	Example

public void addCookie(Cookie cookie)	The method adds the specified cookie to the response. This method can be called more than once to set more than one cookies.	Cookie MyCookie = new Cookie("RollNumber", "156");
		MyCookie.setMaxAge(60*60*24*7*26);
		response.addCookie(MyCookie);

public void sendRedirect(java.lang.String location)throws java.io.IOException)	The method sends a temporary redirect response to the client using the specified redirect location URL. This method can also accept relative URLs.	response.sendRedirect("myserver.com/thePage.htm?ID=737");
public java.lang.String encodeURL(java.lang.String url)	The method is used to encode the specified URL by including session id. This form of the url can be used in html tags that use a url, such as <a href.........>. This enables the server to keep track of the session.	response.encodeURL("buyerPage.jsp");

public java.lang.String encodeRedirectUrl(java.lang.String url)	The method returns a rewritten url that can be used with the sendRedirect method of response object.	if (name == null){
		response.sendRedirect(response.encodeRedirectURL("homePage.jsp"))
		}

image-128.png

Out object

represents the output stream | will be sent to the client as a response to the request

notables

- is an instance of the `jakarta.servlet.jsp.JspWriter` class
- *It uses all standard write(), print(), and println() methods defined in jakarta.servlet.jsp.JspWriter class to display the output. It has page scope.*
- you generally dont have to explicitly call `out.close()`, `out.flush()`, and `out.clear()` as they either are implicitly called within the container/file holding them, or they are pseudo 'legacy' as there are better options to handling their functionality

Scope Communication Objects | Explicit Objects

This is basically a section looking at the various access scopes that can be used in an Application

- **Session** - Anything in this object can be accessed at any JSP so long as the sesion remains *validated*
- **Application** - Anything in this object can be accessed at any JSP in the app
- **Page Context** - Any data declared with this can be accessed at any point within the JSP page

Servlet Objects

notables

- Page object points to the current instance of the JSP page
- Config objects contain initialization parameters used by the Servlet or JSP during its lifecycle.
- Exception Object - this has the same functionality that the error middleware in Express

Use of Standard Actions

notables

- they require the `<jsp>` prefix
- attributes are case-sensitive
- values must be enclosed in double-quotes
- Standard actions can be either an empty or container tag

Various actions *Run at Runtime*

- `<jsp: include>` - allow the inclusion of dynamic content, typically dynamic JSPs etc
- `<jsp: forward>` - Forwards the request object the target page
- `<jsp: param>` - Typical setting of a parameter, mostly used in conjunction with the req object
- `<jsp: text>` - Fairly useless outside of the need to keep the text whitespace exactly as you input it

Checkup

- *C*
- *C* - remember that the request implicit object provides methods to manage the response genereated by the JSP
- B - format this correctly pls
- A - remember session timeout
- *C*
- C