JSP - IMPLICIT OBJECTS

JSP Implicit Objects are the Java objects that the JSP Container makes available to developers in each page and developer can call them directly without being explicitly declared. JSP Implicit Objects are also called pre-defined variables.

JSP supports nine Implicit Objects which are listed below:

Object	Description
request	This is the **HttpServletRequest** object associated with the request.
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.
application	This is the **ServletContext** object associated with application context.
config	This is the **ServletConfig** object associated with the page.
pageContext	This encapsulates use of server-specific features like higher performance **JspWriters**.
page	This is simply a synonym for **this**, and is used to call the methods defined by the translated servlet class.
Exception	The **Exception** object allows the exception data to be accessed by designated JSP.

The request Object:

The request object is an instance of a javax.servlet.http.HttpServletRequest object. Each time a client requests a page the JSP engine creates a new object to represent that request.

The request object provides methods to get HTTP header information including form data, cookies, HTTP methods etc.

We would see complete set of methods associated with request object in coming chapter: JSP - Client Request.

The response Object:

The response object is an instance of a javax.servlet.http.HttpServletResponse object. Just as the server creates the request object, it also creates an object to represent the response to the client.

The response object also defines the interfaces that deal with creating new HTTP headers. Through this object the JSP programmer can add new cookies or date stamps, HTTP status codes etc.

We would see complete set of methods associated with response object in coming chapter: JSP - Server Response.

The out Object:

The out implicit object is an instance of a javax.servlet.jsp.JspWriter object and is used to send content in a response.

The initial JspWriter object is instantiated differently depending on whether the page is buffered or not. Buffering can be easily turned off by using the buffered='false' attribute of the page directive.

The JspWriter object contains most of the same methods as the java.io.PrintWriter class. However, JspWriter has some additional methods designed to deal with buffering. Unlike the PrintWriter object, JspWriter throws IOExceptions.

Following are the important methods which we would use to write boolean char, int, double, object, String etc.

Method	Description
out.print_dataTypedt_	Print a data type value
out.println_dataTypedt_	Print a data type value then terminate the line with new line character.
out.flush	Flush the stream.

The session Object:

The session object is an instance of javax.servlet.http.HttpSession and behaves exactly the same way that session objects behave under Java Servlets.

The session object is used to track client session between client requests. We would see complete usage of session object in coming chapter: JSP - Session Tracking.

The application Object:

The application object is direct wrapper around the ServletContext object for the generated Servlet and in reality an instance of a javax.servlet.ServletContext object.

This object is a representation of the JSP page through its entire lifecycle. This object is created when the JSP page is initialized and will be removed when the JSP page is removed by the jspDestroy method.

By adding an attribute to application, you can ensure that all JSP files that make up your web application have access to it.

You can check a simple use of Application Object in chapter: JSP - Hits Counter

The config Object:

The config object is an instantiation of javax.servlet.ServletConfig and is a direct wrapper around the ServletConfig object for the generated servlet.

This object allows the JSP programmer access to the Servlet or JSP engine initialization parameters such as the paths or file locations etc.

The following config method is the only one you might ever use, and its usage is trivial:

```
config.getServletName();
```

This returns the servlet name, which is the string contained in the <servlet-name> element defined in the WEB-INF\web.xml file

The pageContext Object:

The pageContext object is an instance of a javax.servlet.jsp.PageContext object. The pageContext object is used to represent the entire JSP page.

This object is intended as a means to access information about the page while avoiding most of

the implementation details.

This object stores references to the request and response objects for each request. The application, config, session, and out objects are derived by accessing attributes of this object.

The pageContext object also contains information about the directives issued to the JSP page, including the buffering information, the errorPageURL, and page scope.

The PageContext class defines several fields, including PAGE_SCOPE, REQUEST_SCOPE, SESSION_SCOPE, and APPLICATION_SCOPE, which identify the four scopes. It also supports more than 40 methods, about half of which are inherited from the javax.servlet.jsp. JspContext class.

One of the important methods is **removeAttribute**, which accepts either one or two arguments. For example, pageContext.removeAttribute *"attrName"* removes the attribute from all scopes, while the following code only removes it from the page scope:

```
pageContext.removeAttribute("attrName", PAGE_SCOPE);
```

You can check a very good usage of pageContext in coming chapter: JSP - File Uploading.

The page Object:

This object is an actual reference to the instance of the page. It can be thought of as an object that represents the entire JSP page.

The page object is really a direct synonym for the **this** object.

The exception Object:

The exception object is a wrapper containing the exception thrown from the previous page. It is typically used to generate an appropriate response to the error condition.

We would see complete usage of this object in coming chapter: JSP - Exception Handling.