Java Server Pages (JSP)

Building Dynamic Web Content with Java Enterprise

Introducing Java Server Pages

What is JSP?

Java Server Pages (JSP) is a server-side technology used to create dynamic web pages. It enables developers to insert Java code into HTML pages using special JSP tags, making it easier to separate the user interface from the business logic.

Role in Java EE

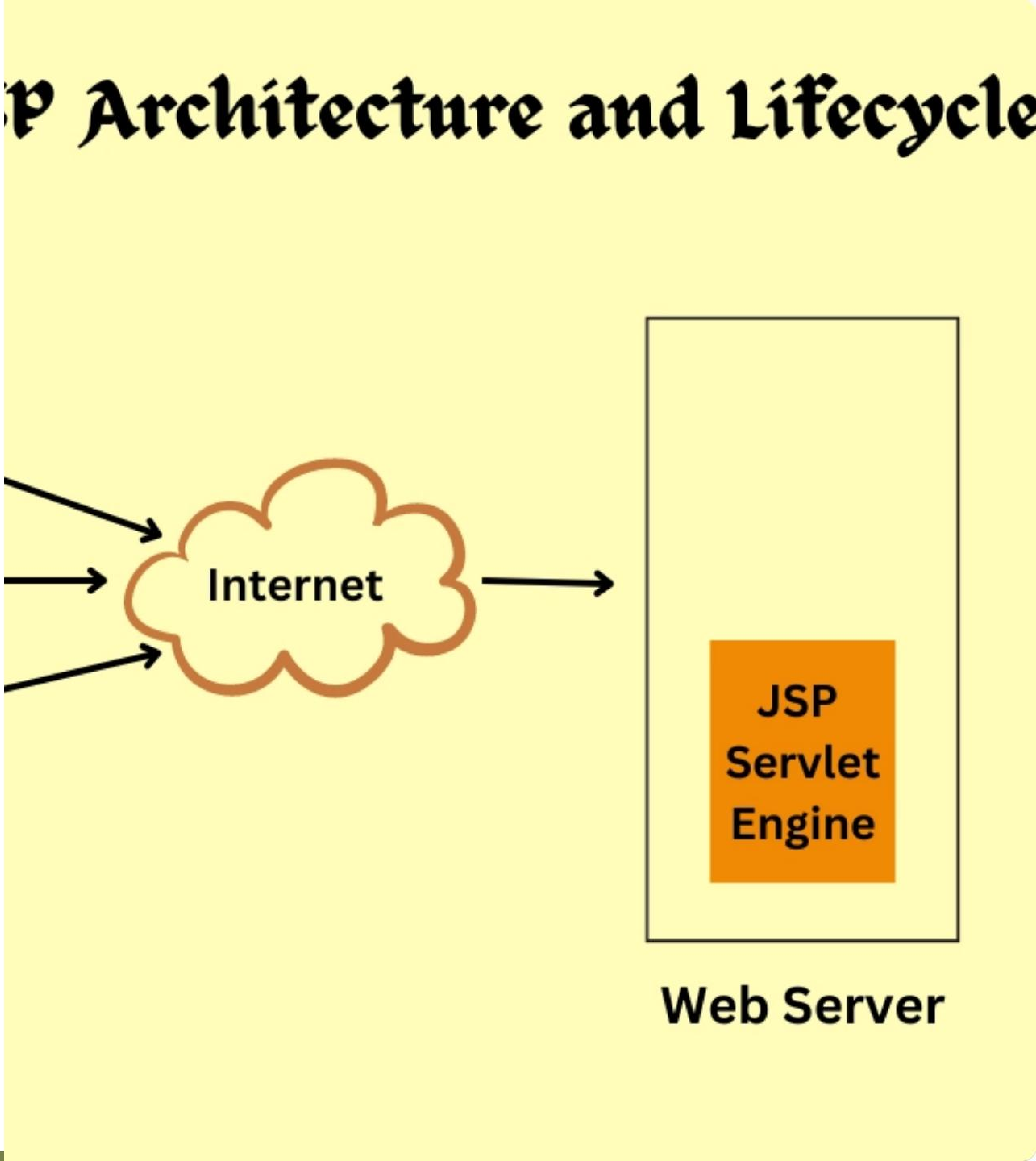
JSP is an integral part of the Java EE (Enterprise Edition) platform. It serves as the **View** component in the MVC (Model-View-Controller) architecture, responsible for presenting data to the user while Servlets handle the processing.

JSP Overview & Lifecycle

Lifecycle Phases

A JSP page goes through several phases before responding to a request:

- **Translation:** JSP is converted into a Servlet.
- **Compilation:** Servlet is compiled into bytecode.
- **Initialization:** `jspInit()` is called.
- **Execution:** `_jspService()` handles requests.
- **Cleanup:** `jspDestroy()` releases resources.



Setting Up the JSP Environment

Java Development Kit (JDK)

Ensure the JDK is installed and the `JAVA_HOME` environment variable is set correctly to compile the generated Servlets.

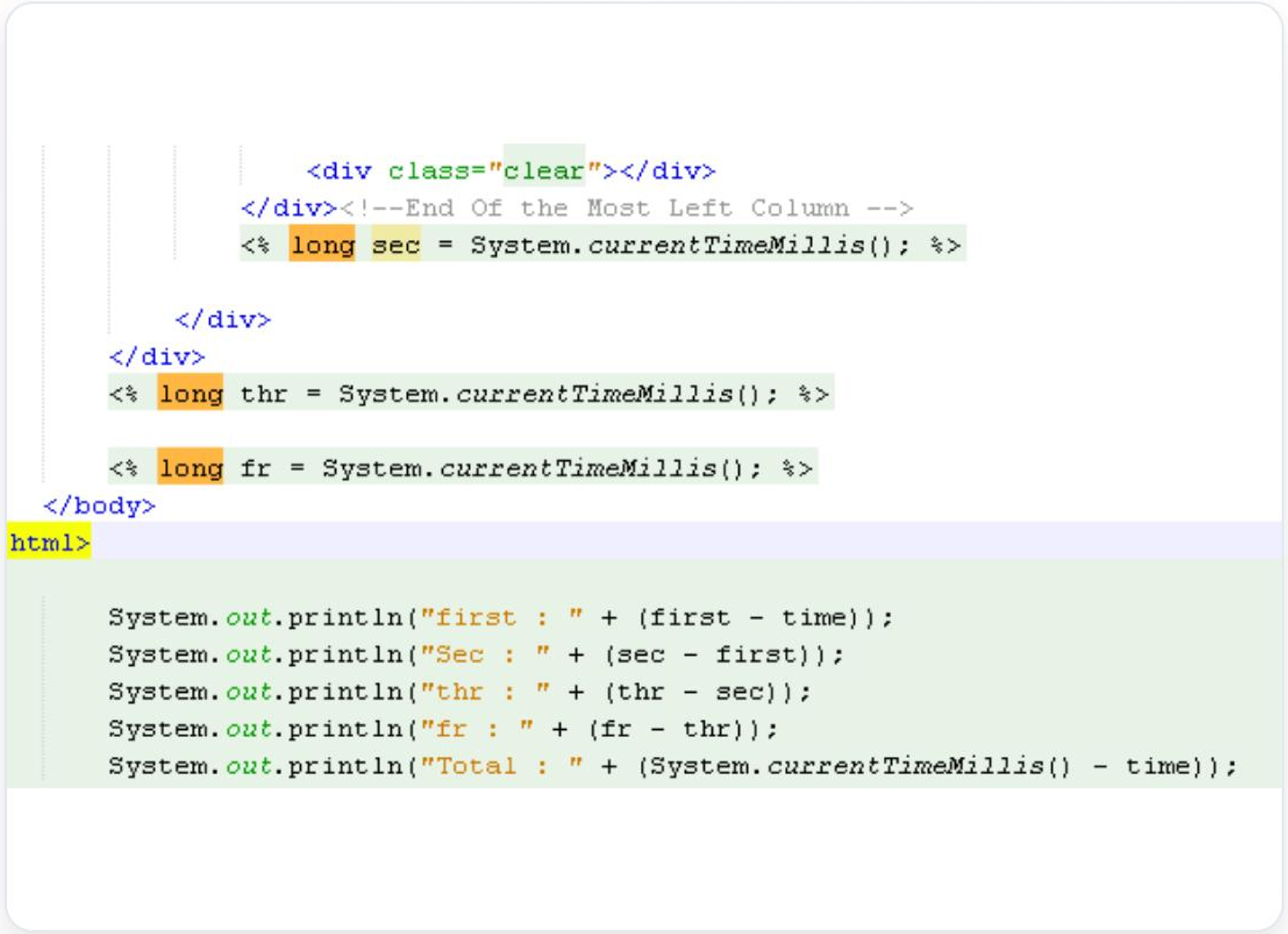
Web Server

A web container like **Apache Tomcat** is required to translate and execute JSP pages. It acts as the JSP engine.

Directory Structure

Place JSP files in the web root. Use `WEB-INF` for configuration files (`web.xml`) and classes that shouldn't be publicly accessible.

Generating Dynamic Content



```
                <div class="clear"></div>
            </div><!--End Of the Most Left Column -->
            <% long sec = System.currentTimeMillis(); %>

        </div>
    </div>
    <% long thr = System.currentTimeMillis(); %>

    <% long fr = System.currentTimeMillis(); %>
</body>
html>

    System.out.println("first : " + (first - time));
    System.out.println("Sec : " + (sec - first));
    System.out.println("thr : " + (thr - sec));
    System.out.println("fr : " + (fr - thr));
    System.out.println("Total : " + (System.currentTimeMillis() - time));
```

Scripting Elements

JSP allows mixing Java with HTML using three main scripting elements:

- **Scriptlets** `<% ... %>`: Contain Java code fragments executed during request processing.
- **Expressions** `<%= ... %>`: Evaluate a Java expression and insert the result directly into the output.
- **Declarations** `<%! ... %>`: Declare variables or methods that apply to the entire JSP page class.

Directives & Action Elements

Directives (Compile-time instructions)

`<%@ page ... %>`: Defines page attributes (language, contentType).

`<%@ include ... %>`: Includes a file during the translation phase.

`<%@ taglib ... %>`: Declares a tag library (e.g., JSTL).

Standard Actions (Runtime instructions)

: Includes a response from another resource at runtime.

: Forwards the request to another page.

: Locates or instantiates a JavaBean.

JSP Standard Tag Library (JSTL)

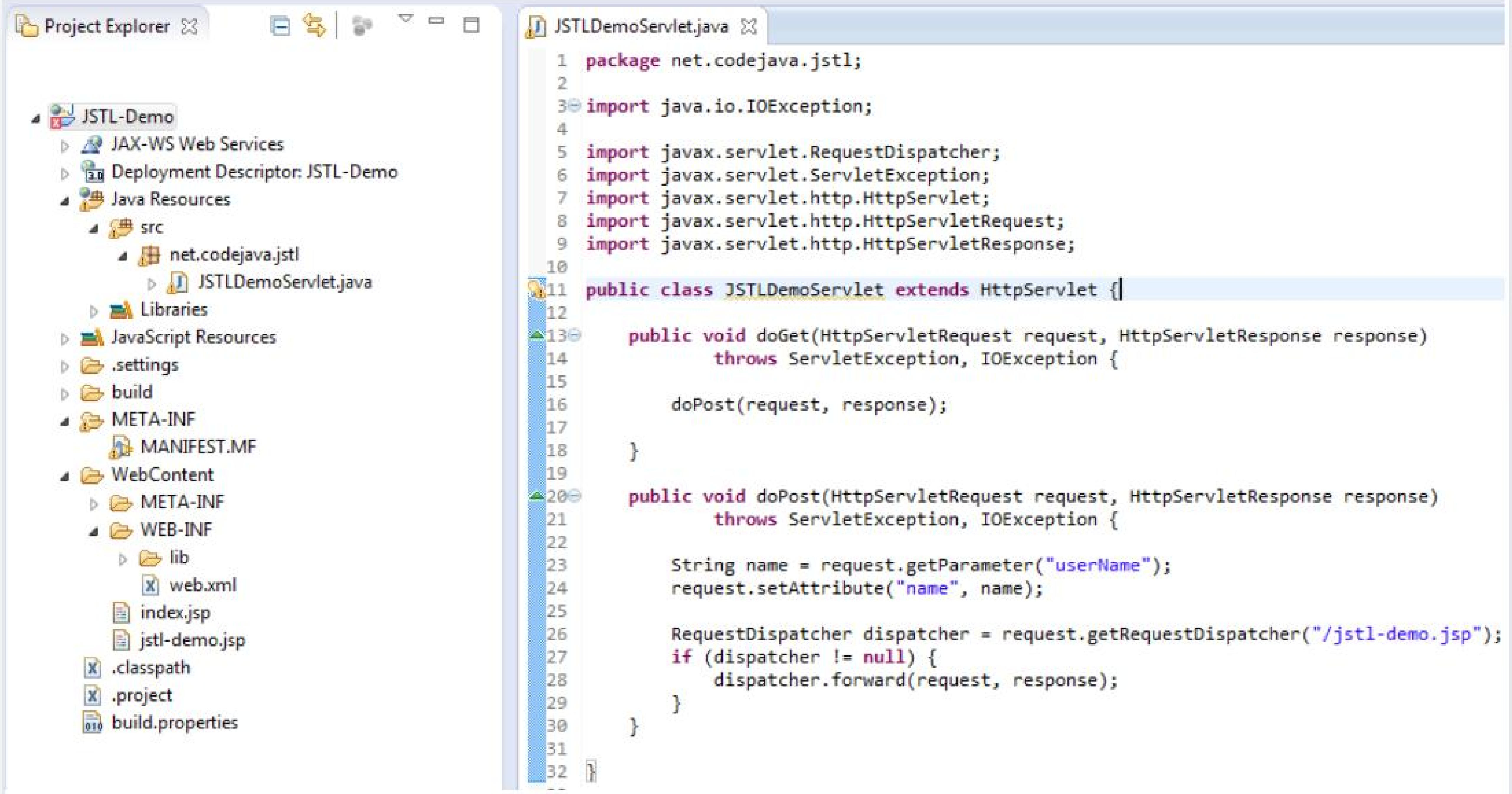
Why use JSTL?

Standard Tag Libraries provide a set of pre-built tags to perform common tasks, reducing the need for scriptlets. This leads to cleaner, more maintainable code.

```
<%-- Avoiding Scriptlets --%>

  Welcome, ${user.name}!
```

It simplifies iteration, conditional logic, and internationalization.



```java
package net.codejava.jstl;

import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class JSTLDemoServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        doPost(request, response);

    }

    public void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        String name = request.getParameter("userName");
        request.setAttribute("name", name);

        RequestDispatcher dispatcher = request.getRequestDispatcher("/jstl-demo.jsp");
        if (dispatcher != null) {
            dispatcher.forward(request, response);
        }
    }

}
```

JSTL Core Tags

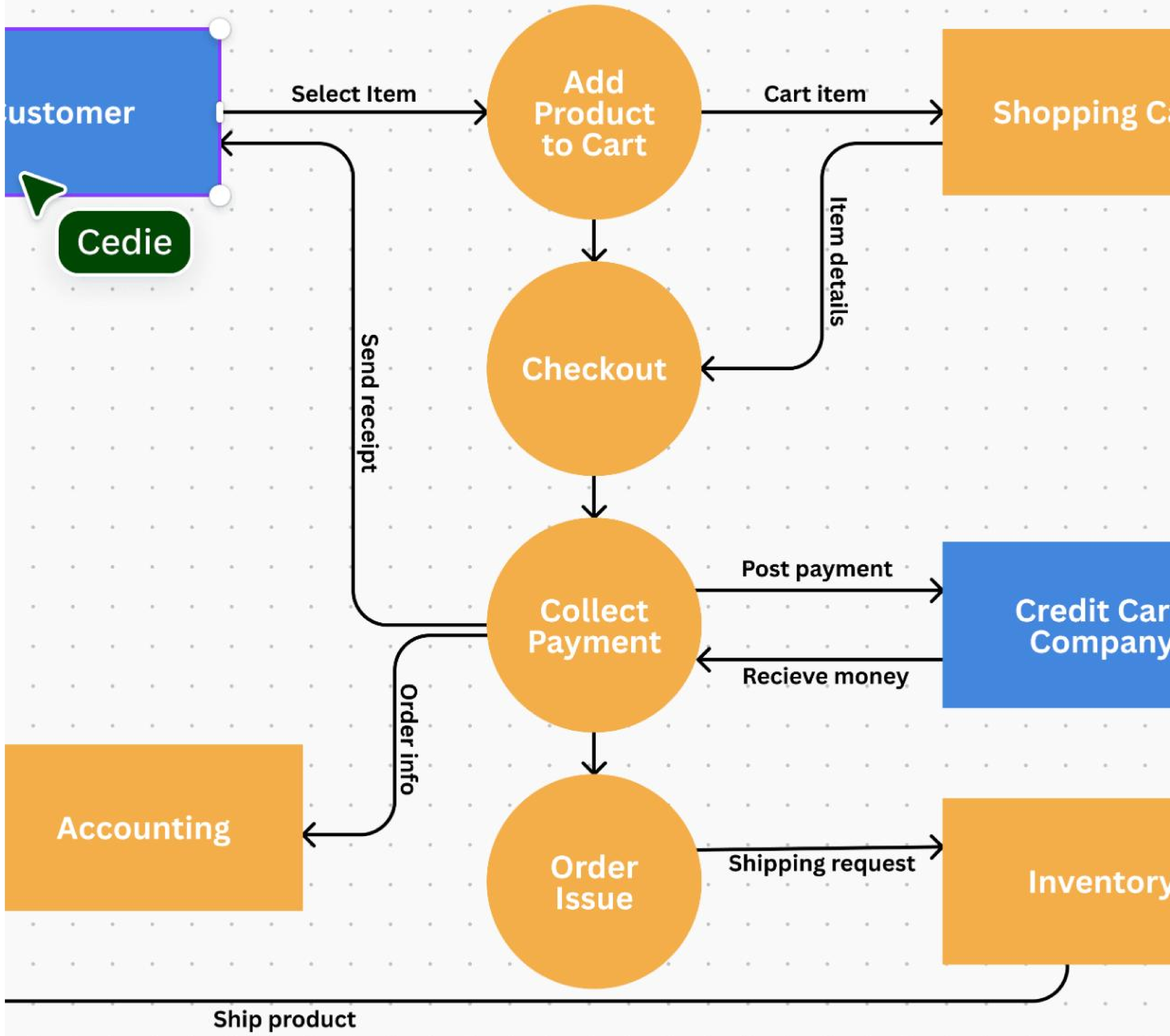
Tag Name	Description	Example Usage
	Outputs a value, similar to an expression.	
	Sets a variable in a specific scope.	
	Conditional execution based on a test.	...
	Iterates over a collection of items.	
	Switch-like structure with when/otherwise.	...

Processing Input

Handling Requests

JSP uses the **Request Object** (`HttpServletRequest`) to capture client input.

- **Form Data:** Use `request.getParameter("name")` to retrieve values from HTML form fields.
- **Multiple Values:** Use `request.getParameterValues()` for checkboxes or multi-select lists.
- **Headers:** Access HTTP headers like User-Agent or Cookies.



Processing Output

Response Object

`HttpServletResponse` is used to modify the response sent to the client. You can set content types, add cookies, or redirect the user.

Out Object

The `JspWriter` "out" object is used to write text directly to the response stream. Example: `out.println("Hello World");`

MIME Types

Control the type of content being generated (e.g., `text/html`, `application/json`) using the page directive or response object.

JSP Implicit Objects Reference

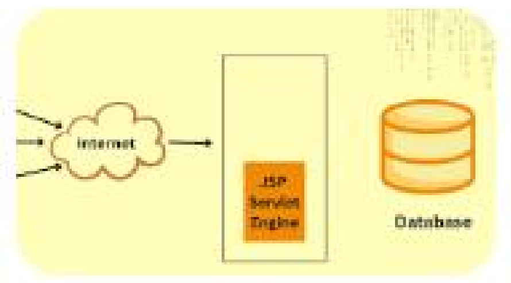
Object	Type	Scope	Purpose
request	HttpServletRequest	Request	Client request data (parameters, headers).
response	HttpServletResponse	Page	Response to the client.
session	HttpSession	Session	User-specific data across multiple requests.
application	ServletContext	Application	Global data shared by all users.
out	JspWriter	Page	Writes content to the output stream.

Other implicit objects: `pageContext`, `config`, `page`, `exception`.

Questions?

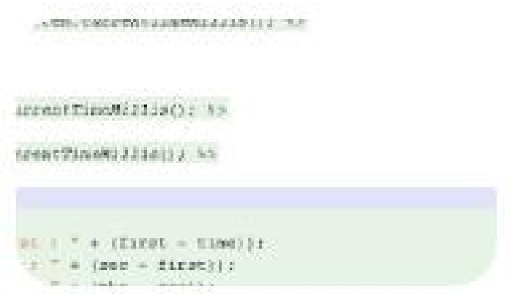
Thank you for your attention.

Image Sources



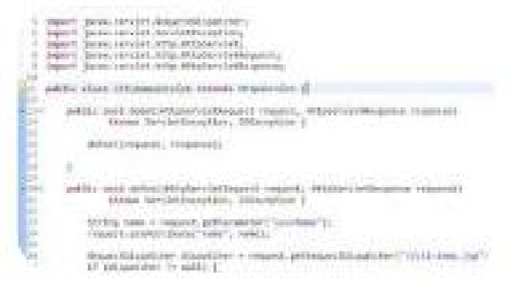
https://files.codingninjas.in/article_images/codingninjas-69-25527.webp

Source: www.naukri.com



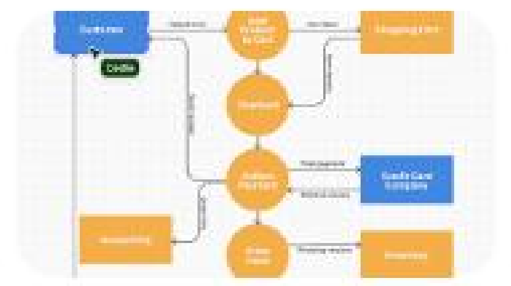
https://i.sstatic.net/p1jT0.png

Source: stackoverflow.com



https://www.codejava.net/images/articles/javaee/jsp/jstl/servlet-compilation-errors-cleared.png

Source: www.codejava.net



https://static-cse.canva.com/blob/1420680/long-form_data-flow-diagram_section-1_asset-1.png

Source: www.canva.com