# JSP y Servlets

**Autor:** José Guillermo Sandoval Huerta

**Fecha:** 15 noviembre 2023

## Índice

# ¿Qué son los Servlets?

Los servlets son módulos de java que sirven para extender las capacidades de los servidores web. Aunque es una definición un poco ambigua los servlets son programas para los servidores. Este sirve de intermediario entre una página JSP y el servidor web donde está alojada la lógica de una aplicación [1].

Un servlet se encargaa de recibir peticiones y request desde un cliente, analiza si es necesario realizar alguna solicitud en particular o brindar una determinada respuesta o response. Para poder tratar cada una de las peticiones, se emplea una serie de métodos dependiendo del verbo por el cual se reciba la petición (GET, POST, PUT, DELETE, etc.).

## Ciclo de Vida del Servlet

- **Inicialización**
  Cuando se despliega en el servidor, un servlet se inicializa llamando al método **init()**. Este método se llama solo una vez durante el ciclo de vida del servlet.

- **Manejo de Solicitudes**
  Cada vez que se realiza una solicitud HTTP al servlet, se llama al método **service()**. Este método determina el tipo de solicitud (GET, POST, etc.) y llama al método correspondiente (doGet(), doPost(), etc.).

- **Destrucción**
  Cuando el servidor decide que un servlet ya no es necesario, se llama al método **destroy()**. Esto ocurre cuando el servidor se apaga o el servlet es retirado del servicio.

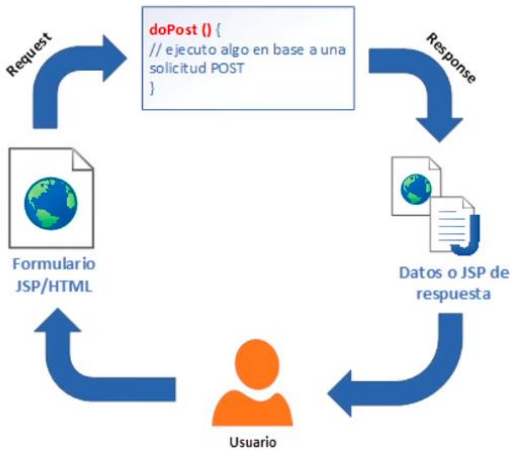


Imagen 1. Ciclo vida de un servlet.

## Métodos de un servlet

Los servlets tienen diferentes métodos que pueden ser utilizados dependiendo del tipo de solicitud que reciban por parte del cliente. Los dos más usados son:

- doGet(): Es el método encargado de recibir las solicitudes que provienen mediante GET.

- doPost(): Es el método encargado de recibir las solicitudes que provienen mediante POST.

Código 1. Ejemplo ejecución Servlet.

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/EjemploServlet")
public class EjemploServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
        // Configuración de la respuesta
        response.setContentType("text/html");

        // Escritura de la respuesta
        PrintWriter out = response.getWriter();
        out.println("<html><body>");
        out.println("<h2>Hola, este es un ejemplo de un Servlet en Java</h2>");
        out.println("</body></html>");
    }
}
```

# ¿Qué son los JSP?

JSP (JavaServer Pages) es una tecnología de Java que se utiliza para desarrollar páginas web dinámicas y basadas en componentes. Está diseñada para simplificar la creación y el mantenimiento de contenido web, permitiendo a los desarrolladores mezclar HTML (o XML) con código Java para generar contenido dinámico.

## Características

1) **Separación de la Lógica y la Presentación**: Una de las características clave de JSP es la capacidad de separar la lógica de presentación del código Java. Esto se logra mediante la inclusión de fragmentos de código Java dentro de las páginas JSP, encerrados en tags especiales <% %>.

**<% String mensaje = "Hola, mundo!"; out.println(mensaje); %>**

2) **Sintaxis Simplificada:** La sintaxis de JSP está diseñada para ser más fácil de entender para los desarrolladores web, ya que se asemeja más a HTML. Esto facilita a los diseñadores web y a los desarrolladores trabajar juntos de manera más eficiente.
3) **Objeto out:** El objeto out se utiliza para imprimir contenido en la respuesta HTTP y se puede acceder directamente en una página JSP.
4) **Tag Libraries (Bibliotecas de Etiquetas):** JSP admite el uso de bibliotecas de etiquetas (Tag Libraries) que proporcionan etiquetas predefinidas para realizar tareas comunes, como bucles y condicionales, directamente en la página JSP. Un ejemplo común es JavaServer Pages Standard Tag Library (JSTL).
5) **Integración con JavaBeans**: JSP se integra fácilmente con JavaBeans, permitiendo la creación y manipulación de objetos Java dentro de las páginas JSP.

**<jsp:useBean id="usuario" class="com.ejemplo.Usuario" />**

**<c:out value="${usuario.nombre}" />**

Código 2. Ejemplo aplicación de JSP.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
   <meta charset="UTF-8">
   <title>Página de Saludo</title>
</head>
<body>
   <h2><%= "Hola, bienvenido a mi página de saludo!" %></h2>
</body>
</html>
```

## ¿Cuándo usar cada uno?

- **Servlets:** Se utilizan cuando se necesita un control más granular sobre la respuesta HTTP y cuando se quiere separar completamente la lógica de presentación y el código de negocio, utilizando, por ejemplo, el patrón de diseño Modelo-Vista-Controlador (MVC).
- **JSP:** Se utilizan cuando se busca una separación más clara entre la lógica de presentación y el código de negocio. Son especialmente útiles para desarrolladores web orientados a la interfaz de usuario que desean trabajar con una sintaxis más parecida a HTML.

## Referencias

[1] J. A. Palos, «Servlets y JSP,» [En línea]. Available: https://lc.fie.umich.mx/~a1039048f/nts/servletsjsp.pdf. [Último acceso: 15 Noviembre 2023].