

Christian Berges Fernández

Índice:

[Servlets. 3](#_Toc465965316)

[Que son y para que se utilizan. 3](#_Toc465965317)

[Ciclo de vida de los servlets. 3](#_Toc465965318)

[Ejemplo de Hola Mundo mediante un servlet. 3](#_Toc465965319)

[JSP( JavaServer Pages). 5](#_Toc465965320)

[Que es y para que se utiliza. 5](#_Toc465965321)

[Principales ventajas. 5](#_Toc465965322)

[Ejemplo de una salida JSP. 5](#_Toc465965323)

[EJB (Enterprise JavaBeans) 7](#_Toc465965324)

[Arquitectura básica EJB. 7](#_Toc465965325)

[Tipos de EJB. 7](#_Toc465965326)

[1- De entidad 7](#_Toc465965327)

[2- EJB de Sesión 8](#_Toc465965328)

[3- EJB Dirigidos por mensaje 8](#_Toc465965329)

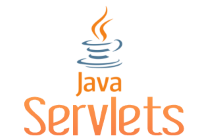
[Funcionamiento. 8](#_Toc465965330)

[Interfaz Home. 8](#_Toc465965331)

[Interfaz Remota. 8](#_Toc465965332)

[Para que es mejor cada una. 9](#_Toc465965333)

[Bibliografía. 9](#_Toc465965334)



# Servlets.

## **Que son y para que se utilizan.**

Los servlets son una clase utilizada en el lenguaje Java que amplía las capacidades del servidor.

Son utilizados para extender las aplicaciones alojadas en los servidores web para que puedan ser vistas como applets de Java que se ejecutan en los servidores en vez de en los navegadores web.

Su uso más extendido es la creación de páginas web dinámicas a partir de los parámetros que se le envíe a través del navegador web.

Sus principales ventajas son:

* Persistencia, siguen “vivos” una vez terminada la petición.
* Eficiencia y menor consumo de recursos.

## **Ciclo de vida de los servlets.**

Se inician el ejecutar el método init del servlet, aunque muchos de ellos se ejecutan en servidores multi-hilos, no tienen tienen de concurrencia durante la inicialización.

Después de la inicialización es cuando los servlets pueden atender las peticiones de los clientes, siendo estas atendidas por la misma instancia del servlet.

Los servlets se ejecutan hasta que el servidor los destruye, bien por cierre del servidor o bien por petición del administrador del sistema, en ambos casos será ejecutando el método destroy del propio servlet.

## **Ejemplo de Hola Mundo mediante un servlet.**

package org.pruebas;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class HolaMundoServlet extends HttpServlet {

/\*\*

\* Servlet de ejemplo que procesa una petición GET

\* @param request

\* @param response

\* @throws ServletException

\* @throws IOException

\*/

@Override

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

response.setContentType("text/html");

PrintWriter out = response.getWriter();

out.println("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 Transitional//EN\">");

out.println("<html>");

out.println("<head><title>Ejemplo Hola Mundo</title></head>");

out.println("<body>");

out.println("<h1>¡Hola Mundo!</h1>");

out.println("</body></html>");

}

}



# JSP( JavaServer Pages).

## **Que es y para que se utiliza.**

Es una tecnología creada para ayudar a crear páginas web dinámicas basadas en HTML, XML y otros tipos de lenguajes.

Para poder desplegar y ejecutar JSP, se requiere un servidor web compatible con servlets.

## **Principales ventajas.**

Al ser de propósito general no se queda simplemente en el mundo web sino que también es apto para crear clases que manejen lógica de negocio y acceso a datos de una manera minuciosa, lo que le permite separar las aplicaciones web en niveles dejando en el archivo JSP la parte encargada de generar el documento HTML.

Otra ventaja en la herencia de la portabilidad de Java, es decir, posibilita la ejecución de las aplicaciones en multiples plataformas sin necesidad de cambios.

## **Ejemplo de una salida JSP.**

package jsp\_servlet;

import java.util.\*;

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.jsp.\*;

import javax.servlet.jsp.tagext.\*;

import com.foo.bar; //importado como resultado de <%@ page import="com.foo.bar" %>

import ...

class \_myservlet implements javax.servlet.Servlet, javax.servlet.jsp.HttpJspPage {

//insertado como

//resultado de <%! int serverInstanceVariable = 1;%>

int serverInstanceVariable = 1;

...

public void \_jspService( javax.servlet.http.HttpServletRequest request,

javax.servlet.http.HttpServletResponse response )

throws javax.servlet.ServletException,

java.io.IOException

{

javax.servlet.ServletConfig config = ...;//obtener la configuración del servlet

Object page = this;

PageContext pageContext = ...;//obtener el contexto de la página para esta petición

javax.servlet.jsp.JspWriter out = pageContext.getOut();

HttpSession session = request.getSession( true );

...

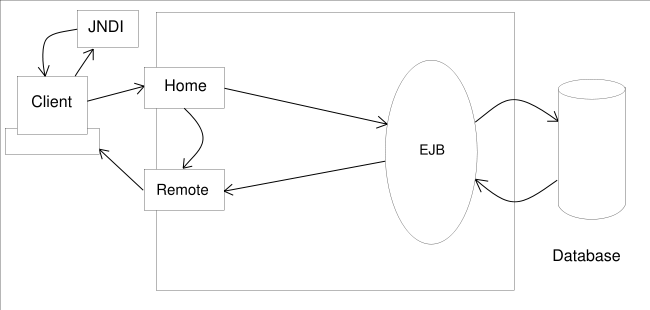


# EJB (Enterprise JavaBeans)

Son una de las interfaces de programación de aplicaciones que forman parte del estándar de construcción de aplicaciones empresariales.

Su principal objetivo es dotar al programador de un modelo que solucione los problemas generales de las aplicaciones empresariales para centrarse en el desarrollo de la lógica de negocio.

## **Arquitectura básica EJB.**



## **Tipos de EJB.**

1. **De entidad**: usados para encapsular los objetos del lado del servidor que almacena los datos, este tipo de EJB presentan la característica fundamental de persistencia:
   1. Persistencia gestionada por el contenedor: el contenedor se encarga de almacenar y recuperar los datos de entidad.
   2. Persistencia gestionada por el vean: el propio objeto entidad se encarga de almacenar y recuperar los datos a los que se refiere.
2. **EJB de Sesión**: gestionan el flujo de la información en el servidor, puede haber dos tipos:
   1. Con estado: las variables de instancia del vean almacenan datos específicos obtenidos durante la conexión con el cliente.
   2. Sin estado: los beans de sesión sin estado son objetos distribuidos que carecen de estado asociado permitiendo que se los acceda concurrentemente.
3. **EJB Dirigidos por mensaje** (Message-Driven EJB): son los únicos beans con funcionamiento asíncrono.

## **Funcionamiento.**

Cada EJB facilita una clase de implementación y dos interfaces java. Las dos interfaces, home y remota, especifican las firmas de los métodos remotos del EJB. Estos se dividen en dos grupos:

* Métodos que no están ligados a una instancia específica, se declaran en la interfaz home y son aquellos utilizados para crear una instancia EJB o para encontrar una entidad EJB existente.
* Métodos ligados a una instancia específica, se ubican en la interfaz remota.

Al ser simples interfaces de java, el contenedor EJB genera clases para las interfaces que actuaran como proxies en el cliente. El cliente llama a un método en los proxies y estos a su vez sitúan los argumentos del método en un mensaje que envía al servidor EJB.

El servidor llamara al método correspondiente a una instancia de la clase de implementación Java para manejar la llamada del método remoto.

### **Interfaz Home.**

Permite al código cliente manipular métodos de clase del EJB que no están asociados a ninguna instancia. Esta interfaz permite crear instancias tanto de entidad como de sesión a través del método créate que puede ser sobrecargado.

### **Interfaz Remota.**

Especifica los métodos de instancia públicos encargados de realizar las operaciones.

Una sesión puede implementar una sola interfaz por un tipo de cliente diferente. La interfaz local es para los clientes que están en la misma máquina virtual que el contenedor EJB y la interfaz remota es para los clientes que están fuera del contenedor del EJB.

# Para que es mejor cada una.

Para realizar aplicaciones web de empresas la mejor de ellas son los Enterprise JavaBeans ya que están específicamente hechos para esa función.

Para el resto de aplicaciones es indiferente usar Servlets o JSP puesto que ambas se desenvuelven de manera muy similar.

# Bibliografía.

Servlets-> <https://es.wikipedia.org/wiki/Java_Servlet>

JSP-> <https://es.wikipedia.org/wiki/JavaServer_Pages>

EJB-> <https://es.wikipedia.org/wiki/Enterprise_JavaBeans>