|  |  |  |
| --- | --- | --- |
| Desarrollo entorno web servidor | octubre 24  2016 | |
|  | | Jorge Sáenz Muñoz |

Servlets

# ¿Qué son? ¿Para qué se utilizan? ¿En qué se diferencian?

Son una clase en el lenguaje de programación Java, utilizada para ampliar las capacidades de un servidor.

Aunque el uso más común de los servlets es generar páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web.

Aunque los Servlets pueden responder a cualquier tipo de solicitudes, éstos son utilizados comúnmente para extender las aplicaciones alojadas por servidores web, de tal manera que pueden ser vistos como applets de Java que se ejecutan en servidores en vez de navegadores web.

Este tipo de Servlets son la contraparte Java de otras tecnologías de contenido dinámico Web, como PHP y ASP.NET.

# Características.

**Manejo de Sesiones:** Se puede hacer seguimiento de usuarios a través de distintos servlets a través de la creación de sesiones.

**Utilización de Cookies:** Las cookies son pequeños datos en texto plano que pueden ser guardados en el cliente. La API de servlets permite un manejo fácil y limpio de ellas.

**Multi-thread:** Los servlets soportan el acceso concurrente de los clientes, aunque hay que tener especial cuidado con las variables compartidas a menos que se utilice la interfaz SingleThreadModel.

**Programación en Java:** Se obtienen las características de multiplataforma o acceso a APIs como JDBC, RMI, etc.

# ¿Para qué tipos de aplicaciones web es mejor? ¿Es indiferente? ¿Por qué? ¿Qué característica diferenciadora tiene?

Los Servlets son un reemplazo efectivo para los CGI en los servidores que los soporten ya que proporcionan una forma de generar documentos dinámicos utilizando las ventajas de la programación en Java como conexión a alguna base de datos, manejo de peticiones concurrentes, programación distribuida, etc.

Por ejemplo, un Servlet podría ser responsable de procesar los datos desde un formulario en HTML como registrar la transacción, actualizar una base de datos, contactar algún sistema remoto y retornar un documento dinámico o redirigir a otro Servlet u alguna otra cosa.

Por ello no es indiferente, y que al tener un procesamiento más rápido que el resto, lo mejor es utilizarlo en aplicaciones web que tengan que procesar datos y transacciones.

**Fuente de información:**

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

<https://users.dcc.uchile.cl/~lmateu/CC60H/Trabajos/jbarrios/general.html>

JavaServer Pages

# ¿Qué son? ¿Para qué se utilizan? ¿En qué se diferencian?

Es una tecnología similar a los Servlets que ofrece una conveniente forma de agregar contenido dinámico a un archivo HTML por utilizar código escrito en Java dentro del archivo utilizando tags especiales que son procesados por el servidor Web antes de enviarlos al cliente.

La posibilidad de usar APIs de Java hacen de JSP una poderosa herramienta de desarrollo ya que se obtiene la ventaja de la programación orientada al objeto, como creación de clases especiales llamadas componentes o Java Beans, independencia de la plataforma propia de la programación en Java, etc.

El uso común es crear páginas web dinámicas basadas en HTML, XML, entre otros tipos de documentos.

La diferencia entre Servlets y JSP es que los Servlets son clases que deben implementar la clase abstracta HttpServlet, en especial el método doGet() o doPost() y deben ser previamente compilados, mientras que los archivos JSP contienen código Java entre código HTML utilizando los símbolos <% y %>. Por esto un archivo JSP debe ser interpretado por el servidor al momento de la petición por parte del usuario.

# Características.

* Conjunta el poder de Java en el servidor y la flexibilidad de HTML en el navegador.
* No sólo se puede utilizar HTML, sino también XML o WML.
* Hace más fácil reusar componentes con JavaBeans los cuales realizan tareas más específicas.
* Su función es saber cómo procesar una solicitud para crear una respuesta.
* Soporta contenido dinámico que refleja las condiciones del mundo real.
* Es más rápido y fácil crear aplicaciones de web
* Capaz de instanciar cualquier clase de Java

# ¿Para qué tipos de aplicaciones web es mejor? ¿Es indiferente? ¿Por qué? ¿Qué característica diferenciadora tiene?

La principal ventaja de JSP frente a otros lenguajes es que el lenguaje Java es un lenguaje de propósito general que excede el mundo web y que es apto para crear clases que manejen lógica de negocio y acceso a datos de una manera prolija.

Esto permite separar en niveles las aplicaciones web, dejando la parte encargada de generar el documento HTML en el archivo JSP.

Otra ventaja es que JSP hereda la portabilidad de Java, y es posible ejecutar las aplicaciones en múltiples plataformas sin cambios. Es común incluso que los desarrolladores trabajen en una plataforma y que la aplicación termine siendo ejecutada en otra.

**Fuente:**

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

<http://aplicaciones-web-lenguajes-programaci.blogspot.com.es/2011/12/jsp.html>

Enterprise JavaBeans

# ¿Qué son? ¿Para qué se utilizan? ¿En qué se diferencian?

Las Enterprise JavaBeans (también conocidas por sus siglas EJB) son una de las interfaces de programación de aplicaciones (API) que forman parte del estándar de construcción de aplicaciones empresariales J2EE (ahora JEE) de Oracle Corporation (inicialmente desarrollado por Sun Microsystems).

Los EJB proporcionan un modelo de componentes distribuido estándar del lado del servidor.

El objetivo de los EJB es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.) para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables.

Hay tres tipos de EJB:

1. **EJB de Entidad:** su objetivo es encapsular los objetos del lado del servidor que almacena los datos, estos presentan la característica fundamental de la persistencia:
   1. **Persistencia gestionada por el contenedor**: el contenedor se encarga de almacenar y recuperar los datos del objeto de entidad mediante el mapeo o vinculación de las columnas de una tabla de la base de datos con los atributos del objeto.
   2. **Persistencia gestionada por el bean:** el propio objeto entidad se encarga, mediante una base de datos u otro mecanismo, de almacenar y recuperar los datos a los que se refiere, por lo cual la responsabilidad de implementar los mecanismos de persistencia es del programador.
2. **EJB de Sesión:** gestionan el flujo de la información en el servidor. Generalmente sirven a los clientes como una fachada de los servicios proporcionados por otros componentes disponibles en el servidor. Puede haber dos tipos:
   1. **Con estado:** en un bean de sesión con estado, las variables de instancia del bean almacenan datos específicos obtenidos durante la conexión con el cliente. Cada bean de sesión con estado, por tanto, almacena el estado conversacional de un cliente que interactúa con el bean. Este estado conversacional se modifica conforme el cliente va realizando llamadas a los métodos de negocio del bean. El estado conversacional no se guarda cuando el cliente termina la sesión.
   2. **Sin estado:** los beans de sesión sin estado son objetos distribuidos que carecen de estado asociado permitiendo por tanto que se los acceda concurrentemente. No se garantiza que los contenidos de las variables de instancia se conserven entre llamadas al método.
3. **EJB Dirigidos por Mensajes:** Son los únicos beans con funcionamiento asíncrono. Usando el Java Messaging System (JMS), se suscriben a un tema (topic) o a una cola (queue) y se activan al recibir un mensaje dirigido a dicho tema o cola. No requieren de su instanciación por parte del cliente.

# Características.

* Permiten el desarrollo de aplicaciones débilmente acopladas.
* Su comportamiento está especificado por interfaces.
* Su funcionamiento se puede configurar en tiempo de despliegue de forma declarativa (descriptor de despliegue).
* Los componentes EJB son multiplataforma.

# ¿Para qué tipos de aplicaciones web es mejor? ¿Es indiferente? ¿Por qué? ¿Qué característica diferenciadora tiene?

Para aplicaciones empresariales J2EE (Java EE), aplicación de negocio.

No es indiferente, ya que lo mejor es utilizar las características modulares de éste para la gran gestión de datos e información, necesarias en las aplicaciones empresariales, y debido a que se encarga de la aplicación de negocio, la cual se encarga de la funcionalidad del servidor.

La característica diferenciadora es la cantidad de módulos que posee para las distintas funciones de la capa de negocio, ya que la especificación describe como el EJB interactúa con su contenedor y cómo el cliente interactúa con la combinación del EJB y el contenedor.

**Fuente:**

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