
WWW
HTML://
Java
WEB
DEVELOPMENT

- **DESARROLLO APLICACIONES WEB JAVA**

1. Introducción Aplicaciones Web

La **Web dinámica** se ha desarrollado desde un sistema de información distribuido hipermedia (HTML) basado en red que ofrecía información estática hasta un conjunto de portales y aplicaciones en Internet que ofrecían un conjunto variado de servicios web.

Para entender el concepto mejor debemos distinguir que es:

- ✓ **Página web:** un documento escrito en lenguaje **HTML** el cual se compone de estilos **CSS** que le dan formato al contenido, **Javascript** que le da efectos, animaciones e interactividad como complemento a las páginas web. Además, las páginas web pueden contener está **archivos multimedia** como imágenes y videos que le dan apoyo al contenido para ayudar al usuario a entender mejor el contenido.
- ✓ **Sitio web:** Es un **conjunto estructurado de páginas web** dentro de un **dominio** que puede ser el nombre de tu entidad por ejemplo: gregoriofer.com. Cada página podría ser una sección de tu sitio web como puede ser el home, blog, servicios, productos, contacto, etc. Muy importante que un sitio web **no forma parte del negocio** sino más bien **es parte de una estrategia de marketing para lograr las ventas**.
- ✓ **Aplicación web:** La gran diferencia de una aplicación web es que esta si **es el negocio** porque a través de ella tú **puedes consumir los servicios que una empresa ofrece** por ejemplo una aplicación de cursos en línea como **emagister**, o una aplicación para gestionar los alumnos de un centro como **Educamos** o una tienda online donde puedas ordenar productos como **Amazon**.

  Una aplicación web se caracteriza porque requiere una **personalización** más específica para el tipo de negocio al cual se está empleando donde dependiendo de las necesidades y complejidad del negocio es necesario utilizar ciertas **librerías, frameworks, bases de datos, servicios de cloud computing, etc.**

  Para crear estas apps **se requiere de un equipo de desarrollo de software** el cual cree una interfaz de usuario para lograr una mejor experiencia de usuario en ese negocio, una vez realizados los diseños el grupo de desarrolladores web profesionales se encargan de escribir el código fuente para maquetar las interfaces de usuario (**frontend**) y también escribir la lógica de la aplicación

(**backend**), es decir, todo el funcionamiento para que se utilice posteriormente la app. Para el desarrollo web utilizaran infinidad de tecnologías web disponibles en el mercado, libres, de código abierto o de pago.

1.2 Evolución Aplicaciones Web Java

La historia del desarrollo de las **aplicaciones web en Java es muy amplia.** Hemos pasado por muchos posibles enfoques y soluciones sobre cómo construir una aplicación web y siempre llegan nuevas ideas que hacen al sector evolucionar.

Las soluciones de *primera generación* incluyeron *el* ***Common Gateway Interface*** **(CGI)**, que es un mecanismo para ejecutar programas externos en un servidor web. **El problema con los scripts CGI es la escalabilidad, se crea un nuevo proceso para cada petición y se consume de recursos del sistema para cada proceso.**

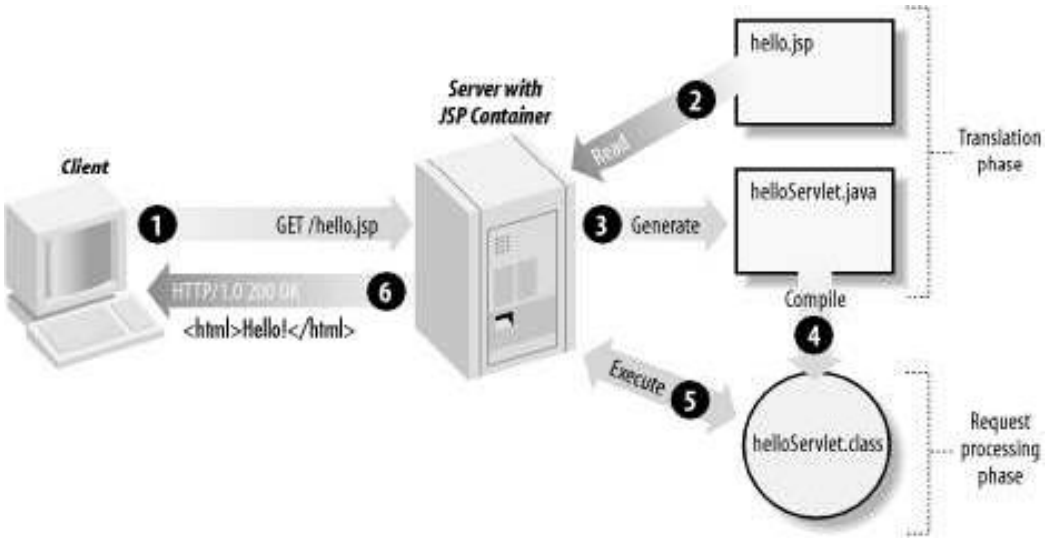
Las soluciones de *segunda generación* incluyeron vendedores de **servidores Web que proporcionaban plug-ins y APIs para sus servidores.** Por ejemplo, una tecnología de segunda generación son los **Servlets**. Los Servlets hacen más fácil escribir aplicaciones del lado del servidor usando la tecnología Java. **El problema con los CGI o los Servlets, sin embargo, es que tenemos que seguir el ciclo de vida de escribir, compilar y desplegar.**

La tecnología Java Server Pages (JSP) son una solución de *tercera generación* que se pueden combinar fácilmente con algunas soluciones de la segunda generación, **creando el contenido dinámico, y haciendo más fácil y más rápido construir las aplicaciones basadas en Web.** Las peticiones de páginas JSP son normalmente implementadas mediante servlets, de forma que el contenedor servlet, al que llamaremos contenedor JSP, **maneja múltiples solicitudes a la vez, y por tanto requiriendo menos recursos.**

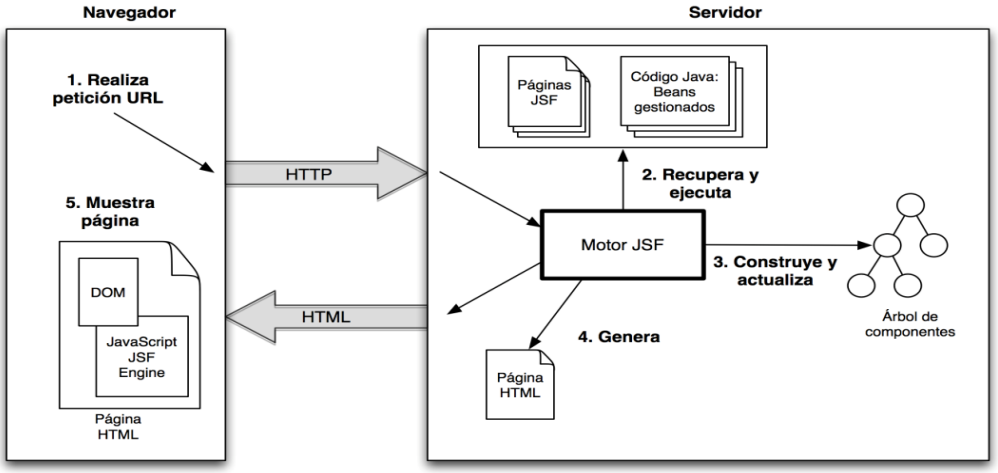
La JSP nos permite poner segmentos de código servlet directamente dentro de una página HTML estática. Cuando el navegador carga una página JSP, se ejecuta el código del servlet y el servidor de aplicaciones crea, compila, carga y ejecuta un servlet en segundo plano para ejecutar los segmentos de código servlet y devolver una página HTML o imprimir un informe XML.

La evolución de las JSP son las JavaServer Faces (JSF). JSF es un Framework MVC basado en el API de Servlets que proporciona un conjunto de **componentes en forma de etiquetas** definidas en páginas

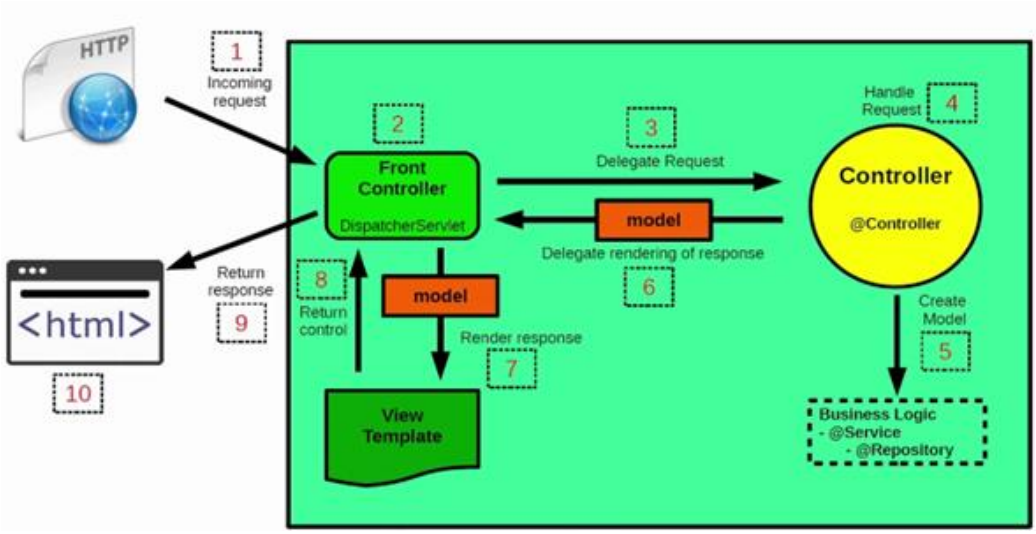
XHTML mediante el **Framework Facelets**. Facelets proporciona características de plantillas y de creación de componentes compuestos creando **mayor rapidez y eficacia** en el desarrollo de la aplicación web. Aunque el estándar es JSF, la evolución y la tendencia actual es **Spring Framework** para desarrollo web que **permite generar paginas HTML sencillas** utilizando otras tecnologías tipo Bootstrap, JQuery Mobile, etc. La diferencia entre JSF y Spring es el nivel de abstracción. Mientras que JSF tiene un nivel de abstracción alto y casi todas las responsabilidades están en el servidor que genera controles de forma automática, Spring es más limpio y genera HTML plano sencillo de tal forma que delega más la responsabilidad en HTML y Javacsript algo que cada día es más tendencia.



Arquitectura web con JSP



Arquitectura web con JSF



Arquitectura web con Spring

1.3 Arquitectura cliente-servidor

Cuando se utiliza un servicio en Internet, también llamado **servicio web**, como consultar una base de datos, transferir un fichero o participar en un foro de discusión, se establece un proceso en el que entran en juego dos partes. Por un lado, **el usuario**, quien ejecuta una aplicación en el ordenador local, el denominado **programa cliente**. Este programa cliente se encarga de ponerse en contacto con el **ordenador remoto** para solicitar el servicio deseado. El ordenador remoto por su parte responderá a lo solicitado mediante un programa que está ejecutando. Este último se denomina **programa servidor.** Los términos **cliente y servidor** se utilizan tanto para referirse a los programas que cumplen estas funciones, como a los ordenadores donde son ejecutados esos programas.

Una de las principales características de los servicios web es que no es necesario que ambas aplicaciones (servidor y cliente) estén escritas en el mismo, lo que hace que la interoperabilidad sea máxima. Por ejemplo, podríamos crear un servicio web en Python y utilizarlo conectándonos desde una aplicación móvil con Android, desde otra aplicación programada en Java o incluso desde otro servicio web escrito con .NET.

Además, **utilizan el protocolo HTTP para el intercambio de información**, lo que significa que la conexión se establece por el **puerto 80** que es el mismo que utilizan los navegadores y que es prácticamente seguro

que **se encuentre abierto en cualquier organización protegida por firewall**. Esto hace que no sea necesario tener especial cuidado abriendo puertos innecesarios para poder conectarnos a ellos.

Cuando pinchamos sobre un enlace hipertexto, en realidad lo que pasa es que establecemos una petición de un archivo HTML residente en el servidor (un ordenador que se encuentra continuamente conectado a la red) el cual es enviado e interpretado por nuestro navegador (el cliente).

Sin embargo, si la página que pedimos no es un archivo HTML, el navegador es incapaz de interpretarla y lo único que es capaz de hacer es salvarla en forma de archivo. Es por ello que, **si queremos emplear lenguajes para realizar un sitio web, es absolutamente necesario que sea el propio servidor quien los ejecute e interprete para luego enviarlos al cliente (navegador) en forma de archivo HTML totalmente legible por él.**

Por lo tanto, podemos hablar de **lenguajes de lado servidor** que son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. Por otro lado, los **lenguajes de lado cliente** son aquellos que pueden ser directamente interpretados por el navegador y no necesitan un pretratamiento.