

# UNIDAD 1: APLICACIONES WEB



## **¿Qué es una aplicación web?**

Una **aplicación web** es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de Internet o de una intranet. Las aplicaciones web son populares debido a la practicidad del navegador web como cliente ligero. La habilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software en miles de potenciales clientes es otra razón de su popularidad. Aplicaciones como los [webmails](#), [wikis](#), [weblogs](#), y tiendas virtuales son ejemplos conocidos de aplicaciones web.

### **Expliquemos su utilidad con una visión previa de las aplicaciones en Internet.**

Pongamos por ejemplo una aplicación de base de datos sobre jurisprudencia y leyes. Nos referimos a un sistema con el cual los abogados pueden revisar distintos fallos judiciales y repasar leyes con la mayor actualización posible. Antes de Internet, sólo podían acceder a estas aplicaciones instalando decenas de diskettes o cd-roms en sus equipos informáticos. A su vez, los usuarios debían instalar actualizaciones constantes para mantenerse al día. También eran necesarios complicados elementos antipiratería (llaves de software, códigos, etc.). A medida que las aplicaciones crecían, también crecían los requerimientos técnicos que debían poseer los equipos informáticos sucediendo muchas veces incompatibilidades y errores de instalación.

Podemos llamar a cada computador utilizado para correr esa aplicación como equipo “cliente”. La aplicación, si bien era desarrollada y distribuida desde un computador o grupo de computadores “maestro”, llegaba a los usuarios en centenas de copias de cds o diskettes.

Con la llegada de Internet y las aplicaciones web, el esquema fue transformándose hasta la actualidad donde sucede lo siguiente: La empresa posee un servidor web (aplicación maestra) a la cual acceden directamente todos los usuarios. Los usuarios poseen claves para identificarse e ingresar. Los usuarios consultan toda la información actualizada al instante directamente ingresando a ese servidor. Los equipos “clientes” sólo necesitan tener un navegador web (Internet Explorer, Firefox, etc.) con mínimos requerimientos técnicos y en cualquier tipo de plataforma operativa (Windows, Linux, Mac OS, Palm, etc. etc.)

Como explicábamos, en este paradigma de cliente-servidor, cada aplicación tenía su propio programa cliente y su interfaz de usuario, estos tenían que ser instalados separadamente en cada estación de trabajo de los usuarios. Una mejora al servidor, como parte de la aplicación, requería típicamente una mejora de los clientes instalados en cada una de las estaciones de trabajo, añadiendo un costo de soporte técnico y disminuyendo la eficiencia del personal.

En contraste, las aplicaciones web generan dinámicamente una serie de páginas en un formato estándar, soportado por navegadores web. Se utilizan lenguajes interpretados del lado del cliente, tales como [JavaScript](#), para añadir elementos dinámicos a la interfaz de usuario. **Generalmente**

cada página web individual es enviada al cliente como un documento estático, pero la secuencia de páginas provee de una experiencia interactiva.

Ver: Cliente – servidor: <http://es.wikipedia.org/wiki/Cliente-servidor>

Por supuesto que este paradigma tiene ciertas limitaciones...

## Interfaz y consideraciones técnicas

**Las interfaces web tienen ciertas limitantes en la funcionalidad del cliente.** Métodos comunes en las aplicaciones de escritorio como dibujar en la pantalla o arrastrar-y-soltar no están soportadas por las tecnologías web estándar. Tampoco se permiten grabar archivos en el computador cliente. Todos los procesos de grabación deben realizarse en el servidor. Imaginen también el esquema informático de los cajeros electrónicos de bancos donde todo el proceso de cálculo y grabado se realiza en los ordenadores centrales.



Una ventaja significativa en la construcción de aplicaciones web que soporten las características de los navegadores o browsers estándar es que deberían funcionar igual independientemente de la versión del sistema operativo instalado en el cliente. En vez de crear clientes para Windows, [Mac OS X](#), [GNU/Linux](#), y otros sistemas operativos, la aplicación es escrita una vez y es mostrada casi en todos lados.



Sin embargo, aplicaciones inconsistentes de HTML, CSS, DOM y otras especificaciones de browsers pueden causar problemas en el desarrollo y soporte de aplicaciones web. Adicionalmente, la habilidad de los usuarios a personalizar muchas de las características de la interfaz (como tamaño y color de fuentes, tipos de fuentes, inhabilitar Javascript) puede interferir con la consistencia de la aplicación web.



Otra (poco común) aproximación es utilizar [Macromedia Flash](#) o [Java applets](#) para producir parte o toda la interfaz de usuario. Como casi todos los browsers incluyen soporte para estas tecnologías (usualmente por medio de plug-ins), aplicaciones basadas en Flash o Java pueden ser implementadas con aproximadamente la misma facilidad. Como hacen caso omiso de las



configuraciones de los browsers, estas tecnologías permiten más control sobre la interfaz, aunque incompatibilidad entre implementaciones de Flash o Java puedan traer nuevas complicaciones. Por las similitudes con una arquitectura cliente-servidor, con un cliente un poco “especializado”, hay disputas sobre si llamar a estos sistemas “aplicaciones web”; un término alternativo es “aplicación enriquecida de Internet”.

## Estructura

Aunque muchas variaciones son posibles, una aplicación web está comúnmente estructurada como una aplicación de tres-capas. En su forma más común, el navegador web es la primera capa, un motor usando alguna tecnología web dinámica (ejemplo: [CGI](#), [PHP](#), [Java Servlets](#) o [ASP](#)) es la capa de en medio, y una base de datos en el servidor como última capa. El navegador web manda peticiones a la capa media, que la entrega valiéndose de consultas y actualizaciones a la base de datos generando una interfaz de usuario.

## Servidor Web

Un **servidor web** es un programa que implementa el *protocolo* [HTTP](#) (*hypertext transfer protocol*). Sin embargo, el hecho de que HTTP y HTML estén íntimamente ligados no debe dar lugar a confundir ambos términos. HTML es un lenguaje de programación y un formato de archivo y HTTP es un protocolo.

Un servidor web se encarga de mantenerse a la espera de *peticiones HTTP* llevada a cabo por un *cliente HTTP* que solemos conocer como *navegador*. El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita. A modo de ejemplo, al teclear <http://www.aprender21.com> en nuestro navegador, éste realiza una petición HTTP al servidor de dicha dirección. El servidor responde al cliente enviando el código HTML de la página; el cliente, una vez recibido el código, lo interpreta y lo muestra en pantalla. Como vemos con este ejemplo, el cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página; el servidor tan sólo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.

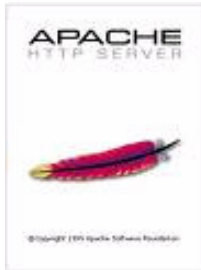
Sobre el servicio web *clásico* podemos disponer de aplicaciones web. Éstas son fragmentos de código que se ejecutan cuando se realizan ciertas peticiones o respuestas HTTP. Hay que distinguir entre:

- Aplicaciones en el lado del cliente: el cliente web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java o Javascript: el servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador, las ejecuta. Es necesario, por tanto, que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas *scripts*). Normalmente, los navegadores permiten ejecutar aplicaciones escritas en lenguaje *javascript* y *java*, aunque pueden añadirse más lenguajes mediante el uso de *plugins*
- Aplicaciones en el lado del servidor: el servidor web ejecuta la aplicación; ésta, una vez ejecutada, genera cierto código HTML; el servidor toma este código recién creado y lo envía al cliente por medio del protocolo HTTP.

Las aplicaciones de servidor suelen ser la opción por la que se opta en la mayoría de las ocasiones para realizar aplicaciones web. La razón es que, al ejecutarse ésta en el servidor y no en la máquina del cliente, éste no necesita ninguna capacidad adicional, como sí ocurre en el caso de querer ejecutar aplicaciones javascript o java. Así pues, cualquier cliente dotado de un navegador web básico puede utilizar este tipo de aplicaciones.

Algunos servidores web importantes son:

- *Apache*
- *IIS*
- *Cherokee*



El **servidor HTTP Apache** es un software (libre) servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Su nombre se debe a que originalmente Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, *a patchy server* (un servidor "parcheado"). **Es el servidor más utilizado en el mundo y es el que utilizaremos para nuestros ejemplos en este curso.**

**Internet Information Services (o Server), IIS**, es una serie de servicios para los ordenadores que funcionan con Windows. Fue integrado en sistemas operativos de Microsoft destinados a ofrecer servicios de red, como Windows 2000 o Windows Server 2003. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS. El servidor web se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas, por ejemplo Microsoft incluye los de Active Server Pages (ASP) y ASP.NET. También pueden ser incluidos los de otros fabricantes, como PHP o Perl.



## **Lenguajes de lado servidor o cliente**

El navegador es una especie de aplicación capaz de interpretar las órdenes recibidas en forma de código HTML fundamentalmente y convertirlas en las páginas que son el resultado de dicha orden.

Cuando nosotros hacemos clic sobre un enlace de hipertexto, en realidad lo que sucede es que establecemos una petición de un archivo HTML residente en el servidor el cual es recibido e interpretado por nuestro navegador (cliente).

Sin embargo, si la página que pedimos no es un archivo HTML, el navegador es incapaz de interpretarla. Es por ello que si se elige un tipo de archivo en otro lenguaje (ej. .php) es necesario que sea el servidor quien ejecute e interprete el archivo y luego envíe al cliente (navegador) un resultado en forma de archivo HTML que sea legible por él.

O sea, cuando elegimos un enlace a una página que contiene un script en un lenguaje comprensible únicamente por el servidor, lo que ocurre es que dicho script es ejecutado por el servidor y el resultado de esa ejecución da lugar a la generación de un archivo HTML que es enviado al cliente.

Los lenguajes de lado servidor son aquellos que son reconocidos, ejecutados e interpretados por el servidor y que se envían al cliente en un formato comprensible para él. Por otro lado, los lenguajes del lado cliente (HTML, Javascript) son aquellos que pueden ser reconocidos, ejecutados e interpretados directamente por el navegador.

## **Introducción a los lenguajes de la WEB**

Tenemos diversos lenguajes para desarrollar nuestras páginas WEB:

### **1. HTML**

- Lenguaje del lado cliente. No permite páginas dinámicas pero puede combinarse con otros lenguajes.
- Es un lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto que es el formato estándar de la web.

### **2. Javascript**

- Lenguaje interpretado del lado cliente. Permite páginas dinámicas. Es rápido y fácil de aprender
- Está basado en el lenguaje Java. Permite realizar tareas y operaciones en el marco de la aplicación cliente sin acceso a funciones de servidor. Se ejecuta en el cliente al tiempo que las sentencias van descargándose junto con el código HTML.
- Por ejemplo podemos crear menús desplegables en nuestras páginas Web.
- Ver [Programación en JavaScript](#)

### **3. VBscript**

- Lenguaje interpretado del lado cliente. Permite páginas dinámicas. Es rápido y fácil de aprender
- Está basado en el lenguaje Visual Basic. Muy similar a Javascript.
- Ver <http://msdn2.microsoft.com/en-us/library/d1wf56tt.aspx>

#### 4. CSS

- Cascading Style Sheets. Hojas de estilo en cascada. Lenguaje de presentación y estructuración de la información de páginas HTML.
- Permite separar la estructura de un documento de su presentación.
- Por ejemplo, el elemento de HTML <H1> indica que un bloque de texto es un encabezamiento y que es más importante que un bloque etiquetado como <H2>. Versiones más antiguas de HTML permitían atributos extra dentro de la etiqueta abierta para darle formato (como el color o el tamaño de fuente). No obstante, cada etiqueta <H1> debía disponer de la información si se deseaba un diseño consistente para una página, y además, una persona que lea esa página con un navegador pierde totalmente el control sobre la visualización del texto. Cuando se utiliza CSS, la etiqueta <H1> no debería proporcionar información sobre como va a ser visualizado, solamente marca la estructura del documento. La información de estilo separada en una hoja de estilo, especifica cómo se ha de mostrar <H1> : color, fuente, alineación del texto, tamaño, y otras características no visuales.
- Ventajas:
  - i. Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
  - ii. Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil, o ser "leída" por un sintetizador de voz.
  - iii. El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.
- Lo veremos más ampliamente en otro curso de este Experto.

#### 5. Applets de Java

- Es otra manera de insertar código a ejecutar en los clientes que visualizan una página web. Son pequeños programas hechos en Java que se ejecutan en el navegador.
- Son aplicaciones precompiladas. No las interpreta el navegador. Permite aplicaciones potentes pero más lentas y tienen un alcance limitado.

#### 6. CGI

- Es el sistema más antigua de programación de páginas dinámicas de servidor. Actualmente está desfasado por su dificultad y alta carga de procesamiento para el servidor.
- Normalmente se escriben en lenguaje Perl o C.

#### 7. Perl

- Es un lenguaje de programación usado para construir aplicaciones CGI. Permite extraer información de archivos de texto y generar informes a partir de su contenido.

## 8. XML

- XML, sigla en inglés de eXtensible Markup Language), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.
- XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable.
- XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

## **Lenguajes de programación web (del lado del servidor)**

Existen numerosos lenguajes de programación empleados para el desarrollo de Aplicaciones Web, entre los que destacan:

- [PHP](#)
- [ASP/ASP.NET](#)
- Java, con sus tecnologías Java Servlets y JavaServer Pages (JSP)
- [Ruby](#)
- [Python](#)



Aunque ciertamente ASP no es un lenguaje de programación, sino una arquitectura de desarrollo web en la que se pueden usar por debajo distintos lenguajes (por ejemplo VB.NET o C# para ASP.NET, o VBScript/JScript para ASP).

### **Consideraciones Técnicas**

Para escribir una página dinámica podemos hacerlo del mismo modo que si lo hiciésemos en HTML. En realidad el código está constituido exclusivamente de texto y lo único que tenemos que hacer es guardar el archivo de texto con una extensión que pueda ser reconocida por el servidor (asp, php, php3).