



Lenguajes de programación en clientes Web

Los sitios web siguen un modelo basado en la programación cliente-servidor con tres elementos comunes:

- **El lado del servidor(server-side):** incluye el hardware y software del servidor Web así como diferentes elementos de programación y tecnologías incrustadas.
- **El lado del cliente(client-side):** este elemento hace referencia a los navegadores web y está soportado por tecnologías como HTML, CSS y lenguajes como JavaScript y controles ActiveX.
- **La red:** describe los diferentes elementos de conectividad utilizados para mostrar el sitio web al usuario.

Por lo tanto, cuando hablamos de *tecnologías empleadas en lenguajes de programación web* podemos citar dos grupos básicos: **client-side** y **server-side**. Las tecnologías client-side son aquellas que son ejecutadas en el cliente, generalmente en el contexto del navegador web. Cuando los programas o tecnologías son ejecutadas o interpretadas por el servidor estamos hablando de programación server-side.

Uno de los objetivos en la programación web es saber **escoger la tecnología correcta** para tu trabajo. Muchas veces los desarrolladores escogen rápidamente una tecnología favorita, que puede ser JavaScript, ColdFusion o PHP y la usan en todas las situaciones. La realidad es que cada tecnología tiene sus pros y sus contras. En general las tecnologías client-side y server-side poseen características que las hacen complementarias más que adversarias. Por ejemplo, cuando añadimos un formulario para recoger información y guardarla en una base de datos, es obvio que tendría más sentido chequear el formulario en el lado del cliente para asegurarnos que la información introducida es correcta, justo antes de enviar la información a la base de datos del servidor. La programación en el lado del cliente consigue que la validación del formulario sea mucho más efectiva y que el usuario se sienta menos frustrado al cubrir los datos en el formulario. Por otro lado el almacenar los datos en el servidor estaría mucho mejor gestionado por una tecnología del lado del servidor (server-side), dando por supuesto que la base de datos estará en el lado del servidor.

Cada tipo general de programación tiene su propio lugar y la mezcla es generalmente la mejor solución. Cuando hablamos de lenguajes de programación en clientes web, podemos distinguir dos variantes:

- Lenguajes que nos permiten dar formato y estilo a una página web (HTML, CSS, etc.).
- Lenguajes que nos permite aportar dinamismo a páginas web (lenguajes de scripting).

En este módulo nos vamos a centrar principalmente en estos últimos, los lenguajes de scripting, y en particular en el lenguaje **JavaScript** que será el lenguaje que utilizaremos a lo largo de todo este módulo formativo.

Tabla comparativa de lenguajes de programación web cliente-servidor

Lado del Cliente (client-side)	Lado del servidor (server-side)
<ul style="list-style-type: none"> ✓ Aplicaciones de Ayuda. ✓ Programas del API del navegador. <ul style="list-style-type: none"> ✦ Plug-ins de Netscape. ✦ Controles ActiveX. ✦ Applets de Java. ✓ Lenguajes de scripting. <ul style="list-style-type: none"> ✦ JavaScript. ✦ VBScript. 	<ul style="list-style-type: none"> ✓ Scripts y programas CGI. ✓ Programas API del servidor. <ul style="list-style-type: none"> ✦ Módulos de Apache. ✦ Extensiones ISAPI y filtros. ✦ Servlets de Java. ✓ Lenguajes de scripting. <ul style="list-style-type: none"> ✦ PHP. ✦ Active Server Pages (ASP/ASP.NET). ✦ ColdFusion. <p>...</p>

Hemos escogido JavaScript por que es el lenguaje de script más utilizado en la programación en el lado del cliente, y está soportado mayoritariamente por todas las plataformas. Por lo tanto a partir de ahora todas las referencias que hagamos estarán enfocadas hacia JavaScript.

A continuación te mostramos un esquema de las **4 capas del desarrollo web** en el lado del cliente, en la que se puede ver que JavaScript se sitúa en la capa superior gestionando el comportamiento de la página web.

Tabla de las 4 capas del desarrollo web en el lado del cliente

Comportamiento (JavaScript)	
Presentación (CSS)	
Estructura (DOM / estructura HTML)	Contenido Estructurado (documento HTML)
Contenido (texto, imágenes, videos, etc.)	

Características

Como vimos anteriormente, los lenguajes de programación para clientes web no son un reemplazo de la programación en el lado del servidor. Cualquier web que reaccione dinámicamente a interacciones del usuario o que almacene datos, estará gestionada por lenguajes de script en el lado del servidor, incluso aunque usemos JavaScript en el cliente para mejorar la experiencia de usuario. Las razones son simples:

- **Primero:** JavaScript por sí mismo no puede escribir ficheros en el servidor. Puede ayudar al usuario a elegir opciones o preparar datos para su envío, pero después de eso solamente podrá ceder los datos al lenguaje de servidor encargado de la actualización de datos.
- **Segundo:** no todos los clientes web ejecutan JavaScript. Algunos lectores, dispositivos móviles, buscadores, o navegadores instalados en ciertos contextos están entre aquellos que no pueden realizar llamadas a JavaScript, o que simplemente son incompatibles con el código de JavaScript que reciben. Aunque ésto ocurra nuestra página web debería ser completamente funcional con JavaScript desactivado. Utilizaremos JavaScript para conseguir que la experiencia de navegación web sea lo más rápida, moderna o divertida posible, pero no dejaremos que nuestra web deje de funcionar si JavaScript no está funcionando.
- **Tercero:** uno de los caminos que más ha integrado la programación cliente con la programación servidor ha surgido gracias a **AJAX**. El proceso "asíncrono" de AJAX se ejecuta en el navegador del cliente y emplea JavaScript. Este proceso se encarga de solicitar datos **XML**, o enviar datos al lenguaje de servidor y todo ello de forma transparente en background. Los datos devueltos por el servidor pueden ser examinados por JavaScript en el lado del cliente, para actualizar secciones o partes de la página web. Es así como funciona una de las webs más populares en Internet, el servicio de Google Maps (<http://maps.google.com>).

JavaScript está orientado a dar soluciones a:

- Conseguir que nuestra página web responda o reaccione directamente a la interacción del usuario con elementos de formulario y enlaces hipertexto.
- La distribución de pequeños grupos de datos y proporcionar una interfaz amigable para esos datos.
- Controlar múltiples ventanas o marcos de navegación, plug-ins, o applets Java basados en las elecciones que ha hecho el usuario en el documento HTML.

- Pre-procesar datos en el cliente antes de enviarlos al servidor.
- Modificar estilos y contenido en los navegadores de forma dinámica e instantáneamente, en respuesta a interacciones del usuario.
- Solicitar ficheros del servidor, y enviar solicitudes de lectura y escritura a los lenguajes de servidor.

Los lenguajes de script como JavaScript no se usan solamente en las páginas web. Los intérpretes de JavaScript están integrados en múltiples aplicaciones de uso cotidiano. Estas aplicaciones proporcionan su propio modelo de acceso y gestión de los módulos que componen la aplicación y para ello comparten el lenguaje JavaScript en cada aplicación. Podríamos citar varios ejemplos como: Google Desktop Gadgets, Adobe Acrobat, Dreamweaver, OpenOffice.org, Google Docs, etc.

Compatibilidad

A diferencia de otros tipos de scripts como los CGI, JavaScript es interpretado por el cliente. Actualmente existen múltiples clientes o navegadores que soportan JavaScript, incluyendo Firefox, Google Chrome, Safari, Opera, Internet Explorer, etc. Por lo tanto, cuando escribimos un script en nuestra página web, tenemos que estar seguros de que será interpretado por diferentes navegadores y que aporte la misma funcionalidad y características en cada uno de ellos. Ésta es otra de las diferencias con los scripts de servidor en los que nosotros dispondremos del control total sobre su interpretación.

Seguridad

JavaScript proporciona un gran potencial para diseñadores maliciosos que quieran distribuir sus scripts a través de la web. Para evitar ésto los navegadores web en el cliente aplican dos tipos de restricciones:

- Por razones de seguridad cuando se ejecuta código de JavaScript éste lo hace en un "espacio seguro de ejecución" en el cuál solamente podrá realizar tareas relacionadas con la web, nada de tareas genéricas de programación como creación de ficheros, etc.
- Además los scripts están restringidos por la política de "mismo origen": la cuál quiere decir que los scripts de una web no tendrán acceso a información tal como usuarios, contraseñas, o cookies enviadas desde otra web. La mayor parte de los agujeros de seguridad son infracciones tanto de la **política** de "mismo origen" como de la política de "espacio seguro de ejecución".

Última modificación: sábado, 9 de septiembre de 2017, 17:36

NAVEGACIÓN



Página Principal

■ Área personal

Páginas del sitio

Curso actual

DEW_B


Participantes

Insignias


General

UD1. Arquitecturas y lenguajes de programación en...

 **U01-C01: Lenguajes de programación en clientes Web**

 U01-C02: Editores de código JavaScript

 U01-C03: Navegadores Web y validador del W3C

 U01-C04: Tecnologías de ejecución en el cliente

 U01-T01: Instalación y configuración de un entorno...

Mis cursos

ADMINISTRACIÓN

