



**XUNTA DE GALICIA**  
CONSELLERÍA DE EDUCACIÓN  
E ORDENACIÓN UNIVERSITARIA

**Ciclo superior:**

**Diseño de aplicaciones web**

**MÓDULO:**

**Desarrollo web en entorno cliente**

Material realizado por Rafael Veiga para el MEC

---

# ÍNDICE DE CONTENIDOS

---

<b>1.</b>	<b><i>Desarrollo Web</i></b>	<b>3</b>
1.1	Áreas	4
<b>2.</b>	<b><i>Lenguajes de programación en clientes web</i></b>	<b>6</b>
2.1	Características	7
2.2	Compatibilidades	8
2.3	Seguridad	9
<b>3.</b>	<b><i>Herramientas y utilidades de programación</i></b>	<b>11</b>
<b>4.</b>	<b><i>Integración de código JavaScript con HTML</i></b>	<b>14</b>
	<b>GLOSARIO</b>	<b>16</b>

---

# ÍNDICE DE TABLAS

---

Tabla 1	Lenguajes del lado de cliente y del lado del servidor	6
Tabla 2	Capas del desarrollo web	6
Tabla 3	ejemplos de editores web gratuitos	11
Tabla 4	Ejemplos de navegadores gratuitos	12

---

# ÍNDICE DE IMÁGENES

---

Ilustración 1	Pirámide con las diferentes facetas en el diseño web	4
Ilustración 2	Icono indicador de la actividad AJAX	7

---

# ÍNDICE DE HERRAMIENTAS

---

Herramienta 1	Herramienta online de validación de estándares del W3C	8
Herramienta 2	Herramienta para ofuscar JavaScript	15

---

# ÍNDICE DE SABER MÁS

---

SABER MÁS 1	Especificaciones HTML y CSS	13
-------------	-----------------------------	----

---

# ÍNDICE DE RECOMENDACIONES

---

Recomendación 1	OTROS EDITORES O ENTORNO DE DESARROLLO	11
-----------------	--	----

---

---

## UNIDAD DIDÁCTICA 1:

# ARQUITECTURAS Y LENGUAJES DE PROGRAMACIÓN EN CLIENTES WEB.

## 1. DESARROLLO WEB

La Web fue inicialmente concebida y creada por **Tim Berners-Lee**, un especialista del laboratorio europeo de partículas (CERN, Consejo Europeo para la Investigación Nuclear) en 1989. En sus mismas palabras, había una *“necesidad de una herramienta colaborativa que soportara el conocimiento científico”* en un contexto internacional. Él y su compañero Robert Cailliau crearon un prototipo web para el CERN y lo mostraron a la comunidad para sus pruebas y comentarios.



Dicho prototipo se basaba en el concepto de . Como resultado se crearon unos y especificaciones que fueron adoptados universalmente e incorporados a Internet y que con aportaciones posteriores como las del NCSA (Centro Nacional de Aplicaciones de Supercomputación), dieron lugar a la popular interfaz MOSAIC.

Todos los prototipos y desarrollos posteriores crecieron bajo la guía del **consorcio W3C** (Consortio World Wide Web), que es una organización con base en el MIT (Instituto Tecnológico de Massachusetts) y que se responsabiliza de desarrollar y mantener los estándares web.

Por **Web se pueden entender tres cosas distintas**: el proyecto inicial del CERN, el conjunto de protocolos desarrollados en dicho proyecto o bien el espacio de información formado por todos los servidores interconectados. Cuando se habla de la Web generalmente se hace referencia a esto último.

Muchas de las discusiones sobre Diseño Web o Desarrollo Web son confusas, ya que la expresión varía considerablemente. Mientras que la mayoría de la gente tiene algún tipo de noción sobre **lo que significa Diseño Web**, solamente unos pocos son capaces de definirlo con exactitud.

Algunos componentes como diseño gráfico o programación forman parte de esa discusión, pero su importancia en la construcción de webs varía de persona a persona y de web a web. Algunos consideran la creación y organización de contenido – o más formalmente, la arquitectura de la información – como el aspecto más importante del Diseño Web. Otros factores como – la facilidad de uso, el valor y funcionalidad del sitio web en la organización, su accesibilidad, publicidad, etc. también forman una parte muy activa hoy en día sobre lo que se considera Diseño Web.

El Desarrollo Web ha sido y sigue estando muy influenciado por múltiples campos como son: las nuevas tecnologías, los avances científicos, el diseño gráfico, la programación, las redes, el diseño de **Interfaz** de usuario, la usabilidad y una variedad de múltiples recursos. Por lo tanto, el Desarrollo Web es realmente un campo multidisciplinar.

## 1.1 ÁREAS

Hay cinco áreas que cubren la mayor parte de las facetas del Desarrollo Web:

- **Contenido:** incluye la forma y organización del contenido del sitio. Esto puede abarcar desde cómo se escribe el texto hasta cómo está organizado, presentado y estructurado usando tecnologías de marcas como HTML.
- **Visual:** hace referencia a la plantilla empleada en un sitio web. Esta plantilla generalmente se genera usando HTML, CSS (Hojas de Estilos en Cascada) (antiguamente incluso Flash) y puede incluir elementos gráficos para decoración o para navegación. El aspecto visual es el aspecto más obvio del Diseño Web, pero no es la única disciplina o la más importante.
- **Tecnología:** aunque muchas de las tecnologías web como HTML o CSS se incluyen en esta categoría, la tecnología en este contexto generalmente hace referencia a los diferentes tipos de elementos interactivos de un sitio web, generalmente aquellos construidos empleando técnicas de programación.
- **Distribución:** la velocidad y fiabilidad con la que un sitio web se distribuye en Internet o en una red interna corporativa está relacionado con el hardware y el software utilizado además del tipo de arquitectura de red utilizada en la conexión.
- **Propósito:** la razón por la que un sitio web existe generalmente está relacionada con algún aspecto de tipo económico. Por lo tanto, este elemento debería considerarse en todas las decisiones que tomemos en las diferentes áreas.

El porcentaje de influencia de cada una de estas áreas en un sitio web, puede variar dependiendo del tipo de sitio que se está construyendo. Una página personal generalmente no tiene las consideraciones económicas que tendría una web que va a vender productos en Internet.

Una forma de pensar en los componentes del Desarrollo Web es a través de la **metáfora de la pirámide** mostrada en la figura. El contenido proporciona los ladrillos que formarán la pirámide, pero la base de la pirámide se fundamenta tanto en la parte visual como en la parte tecnológica y con el punto de vista económico puesto como objetivo o propósito final en la mayoría de los casos.

Aunque la analogía de la pirámide es una forma un poco abstracta de describir el Desarrollo Web, es una herramienta que nos permite visualizar la interrelación de los diferentes componentes de la construcción Web.

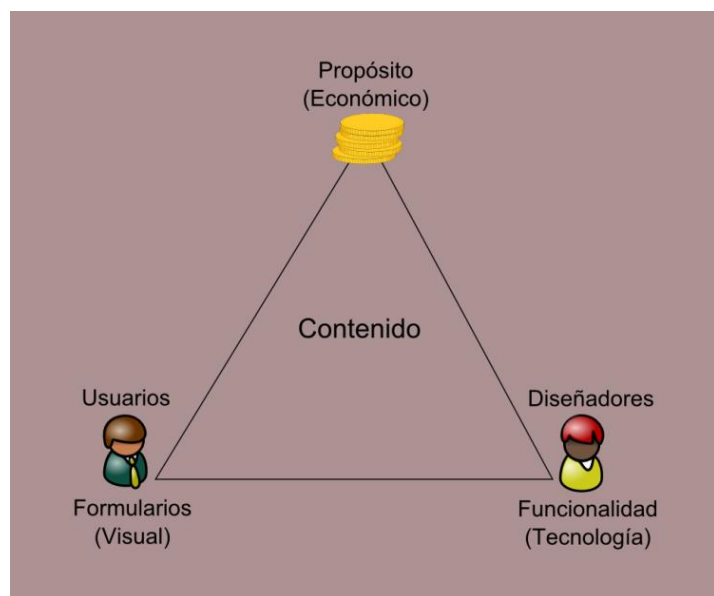


ILUSTRACIÓN 1 PIRÁMIDE CON LAS DIFERENTES FACETAS EN EL DISEÑO WEB

Hoy en día 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. Las tecnologías pueden abarcar un rango amplio desde programas CGI (Interfaz Común de Entrada) escritos en PERL (Lenguaje Práctico para la Extracción e Informes) hasta aplicaciones **Multihilo** basadas en Java, incluyendo tecnologías de servidor de bases de datos que soporten múltiples sitios web.
- 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, los cuales se utilizan para crear la presentación de la página o proporcionar características interactivas. Es justamente aquí dónde nos vamos a centrar a lo largo de todo el módulo.
- La **red**: describe los diferentes elementos de **Conectividad** utilizados para mostrar el sitio web al usuario.

El entendimiento completo de todos los aspectos técnicos del medio Web, incluyendo la componente de red, es de vital importancia para llegar a ser un buen Desarrollador Web.

## 2. LENGUAJES DE PROGRAMACIÓN EN CLIENTES WEB

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 ejecutados o interpretados 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 (Preprocesador Hipertexto 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.

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

TABLA 1 LENGUAJES DEL LADO DE CLIENTE Y DEL LADO DEL SERVIDOR

Hemos escogido JavaScript porque es el lenguaje de **Script** más utilizado en la programación en el lado del cliente, y está soportado mayoritariamente por todas las **Plataforma**. 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.

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

TABLA 2 CAPAS DEL DESARROLLO WEB

## 2.1 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 acciones 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 esto 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 (JavaScript Asíncrono Y XML)**. El proceso “asíncrono” de AJAX se ejecuta en el navegador del cliente y emplea JavaScript. Este proceso se encarga de solicitar datos XML (Lenguaje de Marcas eXtensible), o enviar datos al lenguaje de servidor y todo ello de forma transparente para el usuario (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>).



ILUSTRACIÓN 2 ICONO INDICADOR DE LA ACTIVIDAD AJAX

JavaScript está orientado a dar soluciones:

- 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 las acciones 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.



## 2.2 COMPATIBILIDADES

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. Cada tipo de navegador da soporte a diferentes características del JavaScript y además también añaden sus propios **bugs o fallos**. Algunos de estos fallos son específicos de la plataforma sobre la que se ejecuta ese navegador, mientras que otros son específicos del propio navegador en sí.

A veces las incompatibilidades entre navegadores al interpretar el código de JavaScript no vienen dadas por el propio código en sí, sino que su origen proviene del **Código fuente** HTML. Por lo tanto, es muy importante que tu código HTML siga las **especificaciones del estándar W3C** y para ello dispones de herramientas como el validador HTML del W3C.

<http://validator.w3.org>

### HERRAMIENTA 1 HERRAMIENTA ONLINE DE VALIDACIÓN DE ESTÁNDARES DEL W3C

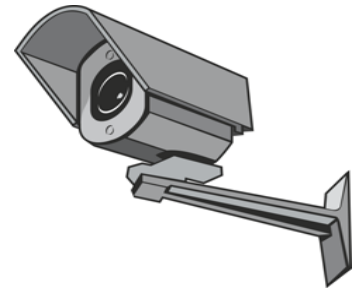
También tienes que tener precaución con las **limitaciones en el uso de JavaScript**:

- No todos los navegadores soportan lenguajes de script (en especial JavaScript) en el lado del cliente.
- Algunos dispositivos móviles tampoco podrán ejecutar JavaScript.
- Incluso las **Implementación** más importantes de JavaScript en los diferentes navegadores no son totalmente compatibles entre ellas como por ejemplo las diferentes incompatibilidades entre Firefox e Internet Explorer.
- La ejecución de código JavaScript en el cliente podría ser desactivada por el usuario de forma manual, con lo que no podremos tener una confianza ciega en que se vaya a ejecutar siempre tu código de JavaScript.
- Algunos navegadores de voz no interpretan el código de JavaScript.

## 2.3 SEGURIDAD

JavaScript proporciona un gran potencial para diseñadores maliciosos que quieran distribuir sus scripts a través de la web. Para evitar esto 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 cual 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 cual 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”.

Al mismo tiempo es importante entender las **limitaciones que tiene JavaScript** y que, en parte, refuerzan sus capacidades de seguridad. JavaScript no podrá realizar ninguna de las siguientes tareas:

- Modificar o acceder a las preferencias del navegador del cliente, las características de apariencia de la ventana principal de navegación, las capacidades de impresión, o a los botones de acciones del navegador.
- Lanzar la ejecución de una aplicación en el ordenador del cliente.
- Leer o escribir ficheros o directorios en el ordenador del cliente (con la excepción de las cookies o los almacenados por el propio código JavaScript).
- Escribir directamente ficheros en el servidor.
- Capturar los datos procedentes de una transmisión en streaming de un servidor, para su retransmisión.
- Enviar e-mails a nosotros mismos de forma invisible sobre los visitantes a nuestra página web (aunque sí que podría enviar datos a una aplicación en el lado del servidor capaz de enviar correos).
- Interactuar directamente con los lenguajes de servidor.
- Las páginas web almacenadas en diferentes dominios no pueden ser accesibles por JavaScript.
- JavaScript es incapaz de proteger el origen de las imágenes de nuestra página.
- Implementar **Multiprocesamiento** o multitarea.
- Otro tipo de **vulnerabilidades** que podemos encontrar están relacionadas con el XSS (Cross-Site Scripting). Este tipo de vulnerabilidad viola la política de “mismo origen” y ocurre cuando un atacante es capaz de inyectar código malicioso en la página web presentada a su víctima. Este código malicioso puede provenir de la base de datos a la cual está accediendo esa víctima. Generalmente este tipo de errores se deben a fallos de implementación de los programadores de navegadores web.

Otro aspecto muy relacionado con la seguridad son los defectos o imperfecciones de los navegadores web o plugins utilizados. Estas imperfecciones pueden ser empleadas por los atacantes para escribir scripts maliciosos que se puedan ejecutar en el sistema operativo del usuario.

El **motor de ejecución de JavaScript** es el encargado de ejecutar el código de JavaScript en el navegador y por lo tanto es en él dónde recaerá el peso fuerte de la implementación de la seguridad. Podríamos citar varios ejemplos de motores de JavaScript:

- Active Script de Microsoft: tecnología que soporta JScript como lenguaje de scripting. A menudo se considera compatible con JavaScript, pero Microsoft emplea múltiples características que no siguen los estándares ECMA (Asociación Europea de Fabricantes de Computadores).
- El kit de herramientas Qt C++ también incluye un módulo intérprete de JavaScript.
- El lenguaje de programación Java en su versión JDK 1.6 (Kit de Desarrollo Java) introduce un paquete denominada `javax.script` que permite la ejecución de JavaScript.
- Y por supuesto todos los motores implementados por los navegadores web como Mozilla, Google, Opera, Safari, etc. Cada uno de ellos da soporte a alguna de las diferentes versiones de JavaScript.

Hoy en día una de las características que más se resalta y que permite diferenciar a unos navegadores de otros, es la rapidez con la que sus motores de JavaScript pueden ejecutar las aplicaciones, y la seguridad y aislamiento que ofrecen en la ejecución de las aplicaciones en diferentes ventanas o pestañas de navegación.

### 3. HERRAMIENTAS Y UTILIDADES DE PROGRAMACIÓN

La mejor forma de aprender JavaScript es tecleando el código HTML y JavaScript en un simple documento de texto. La elección del **Editor** depende de ti, pero aquí te voy a dar algunas pistas para realizar una buena elección.

Para aprender JavaScript no se recomiendan editores del estilo **WYSIWYG** (What You See is What You Get) como Dreamweaver o FrontPage, ya que estas herramientas están más orientadas a la modificación de contenido y presentación, y nosotros nos vamos a centrar más en el código fuente de la página.

Uno de los **factores** más importantes que tienes que tener en cuenta a la hora de elegir un editor, es la facilidad con la que se pueden grabar los ficheros con extensión .html. Independientemente del sistema operativo que estés utilizando cualquier programa que te permita grabar ficheros directamente con la extensión .htm o .html te evitaría un gran número de problemas. También hay que tener en cuenta la **API**. Interfaz de programación de aplicaciones (IPA) o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas (también denominadas vulgarmente "librerías").

**Codificación** que emplea ese programa para grabar los ficheros. Por ejemplo, cuando intentamos guardar archivos con Microsoft Word, el programa intenta almacenarlos en un formato binario de Word – algo que los navegadores web no pueden cargar. Para grabar ese archivo con extensión .txt o .html requiere moverse un poco más por el menú de diálogo “Guardar como”, lo cual es realmente una clara desventaja.

La idea es decantarse por editores que posean **características** que te faciliten la programación web, permitiendo:

- Sintaxis con codificación de colores. Resaltando automáticamente en diferente color o tipo de letra los elementos del lenguaje tales como objetos, comentarios, funciones, variables, etc.
- La verificación de la sintaxis, marcando los errores en la sintaxis del código que estás escribiendo.
- Diferenciar los comentarios del resto del código.
- Generar automáticamente partes del código tales como bloques, estructuras, etc.
- Disponer de utilidades adicionales, tales como cliente FTP (Protocolo de Transmisión de Ficheros) para enviar tus ficheros automáticamente al servidor, etc.

Dependiendo del sistema operativo que utilices en tu ordenador dispones de múltiples **opciones de editores**. Cada una es perfectamente válida para poder programar en JavaScript.

En la siguiente tabla puedes ver algunos ejemplos de editores web gratuitos en función de la plataforma de trabajo:

Plataforma	Editores
Windows	Notepad++, Aptana Studio, Bluefish, Eclipse, NetBeans
Macintosh	Aptana Studio, Bluefish, Eclipse, KompoZer, Nvu
Linux	KompoZer, Amaya, Quanta Plus, Bluefish, codetech

TABLA 3 EJEMPLOS DE EDITORES WEB GRATUITOS

#### RECOMENDACIÓN

Hay otros entornos de desarrollo o editores que están más orientados al desarrollo web como los que puedes consultar en los siguientes enlaces.

**RECOMENDACIÓN 1 OTROS EDITORES O ENTORNO DE DESARROLLO**

Otro de los componentes obligatorio para aprender JavaScript es el **Navegador** web. No es necesario que te conectes a Internet para comprobar tus scripts realizados con JavaScript. Puedes realizar dicha tarea sin conexión. Esto quiere decir que puedes aprender JavaScript y crear aplicaciones con un ordenador portátil y desde cualquier lugar sin necesitar Internet para ello.

El tipo de **navegador web** que utilices es elección tuya. Eso sí, te recomiendo que uses las últimas versiones disponibles para evitar problemas de seguridad e incompatibilidades. En la siguiente tabla puedes ver algunos ejemplos de navegadores gratuitos en función de la plataforma de trabajo:

Plataforma	Navegadores
Windows	Mozilla Firefox, Google Chrome, Safari, Opera, Internet Explorer
Macintosh	Mozilla Firefox, Safari, Google Chrome, Internet Explorer
Linux	Mozilla Firefox, Konqueror, Opera

**TABLA 4 EJEMPLOS DE NAVEGADORES GRATUITOS**

Una recomendación muy interesante es el disponer de 2 ó 3 tipos de navegadores diferentes, ya que así podrás comprobar la compatibilidad de tu página web y ver si tu código fuente de JavaScript se ejecuta correctamente en todos ellos.

Para ajustar un poco más tu entorno de trabajo, lo último que necesitas es el poder ejecutar tu editor web y tu navegador de forma simultánea, ya que el **flujo típico de trabajo en JavaScript** va a ser:



1. Introducir HTML, JavaScript y CSS en el documento original en el editor web.
2. Guardarlo en disco.
3. Cambiarte al navegador web.
4. Realizar una de las siguientes tareas:
  - Si es un nuevo documento, abrirlo a través de la opción Abrir del menú Archivo > Abrir Archivo.
  - Si el documento ya está cargado en el navegador pues simplemente recargar la página.

Los pasos 2 al 4 son acciones que se van a ejecutar frecuentemente. Esa secuencia grabar-cambiar-recargar la realizarás tantas veces cuando estés escribiendo y depurando tu script, que llegará a ser prácticamente un acto reflejo. Algunos editores ya disponen de teclas rápidas para realizar esta tarea. Todo ello dependerá del tipo de editor, navegador y sistema operativo que utilices.

Otro aspecto muy importante es la **validación**. Puedes ahorrarte muchas horas de comprobaciones simplemente asegurándote de que tu código HTML es válido. Si tu código HTML contiene imperfecciones, tienes muchas posibilidades de que tu JavaScript o CSS no funcionen de la manera esperada, ya que ambos dependen de los elementos HTML y sus atributos. Cuanto más te ajustes a las especificaciones del estándar, mejor resultado obtendrás entre los diferentes tipos de navegadores.

El consorcio W3C, el cuál diseñó el lenguaje HTML, desarrolló un validador que te permitirá chequear si tu página web cumple las **Especificación** indicadas por el elemento DOCTYPE que se incluye al principio de cada página web que realices. El Validador será tu amigo y te permitirá realizar unas páginas más consistentes.

Además, ofrece tres formas de introducción de tu código para la validación – copiando y pegando tu código en un formulario, enviando el fichero .html o bien indicando la dirección URL (Localizador Uniforme de Recursos)

dónde se encuentra la página web que se quiere validar. A continuación, se pulsa el botón de chequear y el validador nos devolverá los errores encontrados. Realizaremos los cambios necesarios y procederemos a validar de nuevo hasta que no tengamos más errores.

## PARA SABER MÁS

Seguramente te interesará tener mayor información sobre las diferentes especificaciones HTML y CSS.

Más información sobre [HTML 4.01](#), [HTML5](#) y [XHTML 1.0](#).

Especificación sobre [CSS 2.1](#) y [validador del lenguaje CSS](#)

SABER MÁS 1 ESPECIFICACIONES HTML Y CSS

## 4. INTEGRACIÓN DE CÓDIGO JAVASCRIPT CON HTML

Ahora que ya conoces las herramientas que puedes utilizar para comenzar a programar en JavaScript, vamos a ver la forma de **integrar el código de JavaScript** en tu código HTML.

Los navegadores web te permiten varias opciones de inserción de código de JavaScript. Podremos insertar código usando las etiquetas `<script>` `</script>` y empleando un atributo `type` indicaremos qué tipo de lenguaje de script estamos utilizando:

Por ejemplo:

```
<script type="text/javascript">
// El código de JavaScript vendrá aquí.
</script>
```

**Otra forma de integrar el código de JavaScript** es enlazar un fichero externo que contenga el código de JavaScript. Ésta sería la forma más recomendable, ya que así se consigue una separación entre el código y la estructura de la página web y como ventajas adicionales podrás compartir código entre diferentes páginas, centralizar el código para la depuración de errores, tendrás mayor claridad en tus desarrollos, más modularidad, seguridad del código y conseguirás que las páginas carguen más rápido. La rapidez de carga de las páginas se consigue al tener el código de JavaScript en un fichero independiente, ya que si más de una página tiene que acceder a ese fichero lo cogerá automáticamente de la caché del navegador con lo que se acelerará la carga de la página.

Para ello tendremos que añadir a la etiqueta `script` el atributo `src`, con el nombre del fichero que contiene el código de JavaScript. Generalmente los ficheros que contienen texto de JavaScript tendrán la extensión `.js`.

Por ejemplo:

```
<script type="text/javascript" src="tucodigo.js"></script>
```

Si necesitas cargar más de un fichero `.js` repite la misma instrucción cambiando el nombre del fichero. Las etiquetas de `<script>` y `</script>` son obligatorias a la hora de incluir el fichero `.js`.

**Atención:** no escribas ningún código de JavaScript entre esas etiquetas cuando uses el atributo `src`.

Para **referenciar el fichero origen .js** de JavaScript dependerá de la localización física de ese fichero. Por ejemplo, en la línea anterior el fichero `tucodigo.js` deberá estar en el mismo directorio que el fichero `.html`. Podrás enlazar fácilmente a otros ficheros de JavaScript localizados en directorios diferentes de tu servidor o de tu dominio. Si quieres hacer una referencia absoluta al fichero, la ruta tendrá que comenzar por `http://`, en otro caso tendrás que poner la ruta relativa dónde se encuentra tu fichero `.js`

Ejemplos:

```
<script type="text/javascript" src="http://www.tudominio.com/ejemplo.js"></script>
<script type="text/javascript" src="../js/ejemplo.js"></script>
```

(el fichero `ejemplo.js` se encuentra en el directorio anterior (`../`) al actual, dentro de la carpeta `js/` )

Cuando alguien examine el código fuente de tu página web verá el enlace a tu fichero `.js`, en lugar de ver el código de JavaScript directamente. Esto no quiere decir que tu código no sea inaccesible, ya que simplemente copiando la ruta de tu fichero `.js` y tecleándola en el navegador podremos descargar el fichero `.js` y ver todo el código de JavaScript. En otras palabras, nada de lo que tu navegador descargue para mostrar la página web podrá estar oculto de la vista de cualquier programador.

A veces te puedes encontrar que tu script se va a ejecutar en un **navegador que no soporta JavaScript**. Para ello dispones de una etiqueta `<noscript>Texto informativo</noscript>` que te permitirá indicar un texto adicional que se mostrará indicando que ese navegador no soporta JavaScript.

Si estamos trabajando con **XHTML** (Lenguaje Extensible de Marcado de Hipertexto), la sintaxis para insertar un código de **JavaScript** directamente en el XHTML es un poco diferente:

```
<script type="text/javascript">
  <![CDATA[
    // código de JavaScript a continuación
  ]]>
</script>
```

Los analizadores de XHTML son intolerantes a los elementos que no comienzan por `<` y a las entidades HTML que no comienzan por `&`, y estos caracteres como verás más adelante son ampliamente utilizados en JavaScript. La solución es encapsular las instrucciones de JavaScript en lo que se conoce como sección CDATA:

```
<![CDATA[
    XHTML permite cualquier tipo de carácter aquí incluyendo < y el símbolo &
]]>
```

Como puedes observar el código es un poco complejo en este caso, razón de más para integrar el código de JavaScript en un fichero externo.

Seguramente estarás pensando en cómo puedes **proteger el código de JavaScript** que vas a programar, del uso fraudulento por otros programadores o visitantes a tu página: la respuesta rápida a esa pregunta es que es imposible hacerlo.

Para que el código de JavaScript pueda ejecutarse correctamente deberá ser cargado por el navegador web y por lo tanto su código fuente deberá estar visible al navegador. Si realmente te preocupa que otras personas usen o roben tus scripts, deberías incluir un mensaje de copyright en tu código fuente. Piensa que no solamente tus scripts son visibles al mundo, sino que los scripts del resto de programadores también lo son. De esta forma puedes ver fácilmente cuando alguien está utilizando al pie de la letra tus scripts, aunque esto no evita que alguien copie tu código y elimine tu mensaje de copyright.

Lo más que se puede hacer es ofuscar el código, o hacerlo más difícil de entender. Las técnicas de ofuscación incluyen la eliminación de saltos de línea, espacios en blanco innecesarios, tabuladores, utilización de nombres ininteligibles en las funciones y variables, utilización de variables para almacenar trozos de código, uso de recursividad, etc. La forma más rápida de hacer todas esas tareas de ofuscación es utilizar un software que producirá una copia "**Comprimir**" del script que has programado para facilitar su carga rápida.

Puedes ver un ejemplo de ofuscador de JavaScript en el siguiente enlace:

<http://www.javascriptobfuscator.com/>

## HERRAMIENTA 2 HERRAMIENTA PARA OFUSCAR JAVASCRIPT



Lo mejor es que cambies ese ***¡Error! No se encuentra el origen de la referencia.*** y pienses de una manera diferente. En lugar de proteger tu código, lo mejor es promocionarlo y hacer ostentación de él, añadiendo comentarios útiles en el código fuente, publicándolo en tu blog o en webs de programación, etc. Incluso podrás añadir una licencia **Creative Commons** para animar a la gente a que lo utilice, lo copie y lo mantenga público al resto del mundo. Así



conseguirás mayor reputación como buen programador y la gente contactará contigo para más información, posibles trabajos, etc.

## GLOSARIO

**API.** Interfaz de programación de aplicaciones (IPA) o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas (también denominadas vulgarmente "librerías").

**Codificación.** Representación o método utilizado por las diferentes aplicaciones o programas para almacenar la información.

**Código fuente.** Conjunto de líneas de texto que contienen las instrucciones que el ordenador debe ejecutar.

**Comprimir.** Proceso mediante el cual conseguimos reducir el espacio físico que ocupa el código fuente de nuestra aplicación, sin que se produzca pérdida de información.

**Conectividad.** Capacidad que tiene un dispositivo para poder conectarse a otros. Aquí se detallan los diferentes protocolos y material utilizado para poder realizar dicha conexión.

**Editor.** Programa que nos permite crear y editar documentos en diferentes formatos.

**Especificación.** Documento técnico que establece las normas, formatos, características, etc. a seguir en el desarrollo de un producto o aplicación.

. Es aquel texto de tal forma que, cuando pulsamos en él nos conduce a otro texto, objeto, sonido, video, sección o documento relacionado.

**Implementación.** Es la realización de una especificación técnica en forma de programa, componente u otro sistema de cálculo.

**Interfaz.** Es el medio o la forma a través de la cuál un usuario se comunica con el ordenador.

**Intérprete.** Programa informático capaz de analizar y ejecutar código escrito en un lenguaje de programación de alto nivel (aquel que se acerca más a la capacidad cognitiva de las personas en lugar de a la capacidad ejecutora de la máquina).

**Multihilo.** También denominado multiproceso hace referencia a la posibilidad de ejecutar diferentes trozos de código de una misma aplicación de forma simultánea.

**Multiprocesamiento.** Es la ejecución simultánea de varios trozos de código de una misma aplicación o proceso.

**Navegador.** Programa que permite al usuario visualizar documentos de hipertexto almacenados en servidores web en Internet.

*¡Error! No se encuentra el origen de la referencia..* Modelo o patrón de pensamiento.

**Plataforma.** Sistema operativo utilizado por un determinado dispositivo.

. Conjunto de reglas y normas que usan los ordenadores para poder comunicarse unos con otros.

**Script.** Lenguaje de guiones o lenguaje de órdenes que se almacena por lo general en archivos de texto plano y que será ejecutado por un programa intérprete.