Arquitecturas y lenguajes de programación en clientes web.

Caso práctico

La empresa "**BK Programación**", ha sido informada a través de uno de sus asesores, de que se ha abierto el plazo, para concursar a la adjudicación de un proyecto de modernización de la web de una importante empresa dedicada a la moda.

Ada, la directora de "**BK Programación**", decide que es una buena oportunidad para conseguir trabajo para su empresa, por lo que consultan el pliego de requisitos exigidos y ve que su empresa tiene personal suficiente para dar solución a dicho proyecto. Se envía la solicitud y dos meses más tarde, la empresa de **Ada** sale adjudicataria del contrato de modernización.

El objetivo principal del proyecto es el de dar un mayor dinamismo a las páginas y actualizar la web a lo que se conoce como Web 2.0 (mayor interacción, interoperabilidad, aplicaciones más ricas y no intrusivas, etc.)

Ada considera que **Antonio** (estudiante de ciclo y trabajador de su empresa), con conocimientos de lenguaje HTML, podría formarse en las técnicas necesarias para realizar dicho trabajo de actualización y como además el proyecto tendrá un plazo máximo de entrega de 1 año, dispondrá de tiempo más que suficiente para ello.

Ada, informa a sus trabajadores de la adjudicación de dicho contrato, y **Antonio** (bajo la tutoría de **Juan**), decide comenzar a investigar en las diferentes opciones disponibles para llevar a cabo su trabajo.

En esta unidad de trabajo nos vamos a introducir en los tipos de lenguajes de diseño web en entorno cliente, sus características, sus limitaciones, aspectos de seguridad y veremos algunas herramientas que podrás utilizar para configurar tu entorno de trabajo y comenzar a programar en JavaScript.

¡Pues vayamos a ello!

Materiales formativos de FP Online propiedad del Ministerio de Educación y Formación Profesional.

Aviso Legal

1.- Desarrollo web.

Caso práctico

Juan (técnico en Desarrollo de Aplicaciones Informáticas), será el tutor de **Antonio** en la empresa **BK Programación** y le irá guiando durante todo el proceso de formación y en la realización del proyecto de modernización de la web de la compañía de moda.

Para comenzar, **Juan** le recomienda a **Antonio** una pequeña introducción sobre lo que es el Desarrollo Web y las diferentes áreas que abarca el Diseño Web.

La web fue inicialmente concebida y creada por **Tim Berners-Lee**, un especialista del laboratorio europeo de partículas (CERN) 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.

Silvio Tanaka (CC BY-NC)

Dicho prototipo estaba basado en el concepto de hipertexto. Como resultado se crearon unos protocolos y especificaciones que han sido adoptados universalmente e incorporados a Internet, gracias a aportaciones posteriores como el desarrollo por la NCSA de la popular interfaz MOSAIC.

Todos los prototipos y desarrollos posteriores crecieron bajo la guía del **consorcio W3C**, que es una organización con base en el MIT 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, y tú vas a estar dentro de ese grupo.

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 funcionalidad, 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 el de las nuevas tecnologías, los avances científicos, el diseño gráfico, la programación, las redes, el diseño de interfaces de usuario, la usabilidad y una variedad de múltiples recursos. Por lo tanto el Desarrollo Web es realmente un campo multidisciplinar.

Autoevaluación

¿Cuando nos referimos a la Web estamos hablando de?

- ○ Proyecto del CERN.
- ○ Protocolos utilizados en el proyecto del CERN.
- ○ Espacio de información que nos proporcionan los servidores a través de Internet.

No es correcta, ya que dicho proyecto es el origen de lo que hoy entendemos por la Web.

No es cierto, ya que los protocolos simplemente son un componente más que se utiliza para el buen funcionamiento de las comunicaciones en la Web.

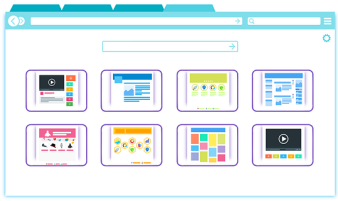
Efectivamente es correcta, ya que hoy en día cuando hablamos de Web estamos hablando de todo ese entramado de información proporcionado por los servidores de información interconectados.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta

1.1.- Áreas.

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



200degrees (Pixabay License)

- **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 CSS 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 entran dentro de 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/software utilizado y el 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 Diseño Web es a través de la metáfora de la pirámide mostrada en la figura anterior. 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 Diseño Web, es una herramienta que nos permite visualizar la interrelación de los diferentes componentes de la construcción Web.

Hoy en día los sitios web siguen un modelo basado en la **programación cliente-servidor** con tres **elementos** comunes:

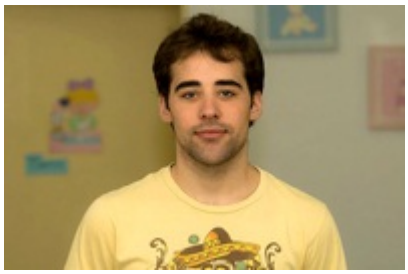
- El lado del **servidor (server-side)**: incluye el hardware y software del servidor Web así como diferentes elementos de programación y tecnologías incrustadas. Las tecnologías pueden abarcar un rango amplio desde programas CGI escritos en PERL hasta aplicaciones multihilo basadas en Java, incluyendo tecnologías de servidor de bases de datos que soporten múltiples sitios web. Hoy día se utiliza mucho en el lado servidor el lenguaje `node.js` basado en JavaScript, pero para el lado del serivodr
- 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 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 Diseñador Web.

2.- Lenguajes de programación en clientes web.

Caso práctico



El proyecto de modernización de la empresa de moda va a consistir, principalmente, en añadir mayor funcionalidad a las páginas ya existentes, más control en los formularios, dinamismo, modernización de la mecánica de varios procesos, etc.

Antonio tendrá que centrarse en los lenguajes de programación en el lado del cliente, ya que éstos son los que le ayudarán a aportar las capacidades solicitadas en el proyecto.

Antonio se pone manos a la obra y consulta bajo la tutoría de **Juan**, qué opciones tienen para llevar a cabo esa programación y analizan las características, compatibilidades y seguridad aportada por los lenguajes de programación en el lado del cliente.

Sara_Torda (Pixabay License)

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 o PHP (generalmente) 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 grabarla 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, aunque siempre también deberemos validar dicha información en el lado del servidor para asegurar la integridad. 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)

- Aplicaciones de Ayuda.
- Programas del API del navegador.
  - Plug-ins de Netscape.
  - Controles ActiveX.
  - Pepper para Chrome.
  - Applets de Java.
- Lenguajes de scripting.
  - JavaScript.
  - VBScript.
	- Scripts y programas CGI.
	- Programas API del servidor.
	  - Módulos de Apache.
	  - Extensiones ISAPI y filtros.
	  - Servlets de Java.
	- Lenguajes de scripting.
	  - PHP.
	  - Active Server Pages (ASP/ASP.NET).
	  - JavaScript (Librería Node.js)

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 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:

1.- Comportamiento (JavaScript).
2.- Presentación (CSS)
3.- Estructura (DOM / estructura HTML)
4.- Contenido (texto, imágenes, vídeos, etc.)

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 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:

Everaldo Coelho (GNU/GPL)

- **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 funcionan hoy día la mayoría de las web.

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.

Autoevaluación

¿Podríamos escribir con JavaScript un fichero de texto directamente en el servidor?

- ○ Sí, pero sólo si se trata de una cookie.
- ○ De ninguna manera.
- ○ Sí, empleando AJAX.

No es correcta, ya que aunque JavaScript puede escribir cookies lo hará en el lado del cliente.

Efectivamente es correcta ya que JavaScript es un lenguaje que interpreta el navegador web y no es un lenguaje de servidor, por lo que no tendrá acceso a los recursos para poder escribir en él.

No es cierto, ya que AJAX lo que nos permitirá es hacer peticiones en background al servidor pero no escribir nada directamente en el servidor puesto que AJAX está basado en JavaScript y es una tecnología en el lado del cliente.

Solución

1. Incorrecto
2. Opción correcta
3. Incorrecto

2.2.- Compatibilidades.

A diferencia de otros tipos de scripts, 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.

Everaldo Coelho (GNU/GPL)

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.

Para saber más

¿Te gustaría ver cómo soporta cada navegador las diferentes características de Javascript y poder verlo por versiones?

Comparación de navegadores web.

A veces las incompatibilidades entre navegadores al interpretar el código de JavaScript no vienen dadas por el propio código en si, 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 W3C:

Validador 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 implementaciones más importantes de JavaScript en los diferentes navegadores no son totalmente compatibles entre ellas: por ejemplo 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.

Everaldo Coelho (GNU/GPL)

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**".

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).
- 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 si 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. 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 cuál 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. Éstas 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

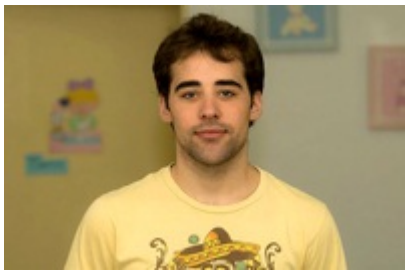
características que no siguen los estándares ECMA.

- El kit de herramientas Qt desarrollado en C++ también incluye un módulo intérprete de JavaScript.
- El lenguaje de programación Java en su versión JDK 1.6 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.

Caso práctico



En **BK Programación Juan** y **Antonio** han decidido, después de analizar los requisitos del proyecto y de consultarlo con su directora **Ada**, que utilizarán el lenguaje JavaScript para la realización del proyecto de modernización de la empresa de moda.

Lo primero que tienen que hacer es decidir qué tipo de herramientas y utilidades adicionales van a usar para realizar la programación con el lenguaje JavaScript. Buscan alguna herramienta que les permita introducir el código de JavaScript fácilmente y les aporte ayudas adicionales detectando errores sintácticos, partes incompletas, etc.

También tienen que decidir qué navegador o navegadores van a usar para comprobar la ejecución y compatibilidad de su código de JavaScript.

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, 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** importantes que tienes que tener en cuenta a la hora de elegir un editor, es ver 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 dicha extensión te evitaría un gran número de problemas. También hay que tener en cuenta la codificación que emplea ese programa para grabar los ficheros.

Hoy día no es necesario contar con un IDE para desarrollar en Javascript. Hay editores ligeros que, a día de hoy, nos ofrecen casi las mismas características. Lo que sí debes tener en cuenta es que te aporten las siguientes características para hacer tu trabajo más fácil:

- **Resaltado de texto.** Muestra con distinto color o tipo de letra los diferentes elementos del lenguaje: sentencias, variables, comentarios, etc. También genera indentado automático para diferenciar de forma clara los distintos bloques de un programa.
- **Completado automático.** Detecta qué estás escribiendo y cuando es posible te muestra distintas opciones para completar el texto.
- **Navegación en el código.** Permite buscar de forma sencilla elementos dentro del texto, por ejemplo, definiciones de variables.

- **Comprobación de errores al editar.** Reconoce la sintaxis del lenguaje y revisa el código en busca de errores mientras lo escribes.
- **Generación automática de código.** Ciertas estructuras, como la que se utiliza para las clases, se repiten varias veces en un programa. La generación automática de código puede encargarse de crear la estructura básica, para que sólo tengas que rellenarla.
- **Gestión de versiones.** En conjunción con un sistema de control de versiones, el entorno de desarrollo te puede ayudar a guardar copias del estado del proyecto a lo largo del tiempo, para que si es necesario puedas revertir los cambios realizados.

Editores más populares para trabajar con javascript

VSCode

Visual Studio Code de Microsoft es una herramienta muy versátil, a la que se le pueden instalar muchas extensiones para darle una gran funcionalidad en casi cualquier lenguaje de programación. Visual Studio Code.

Microsoft Visual Code (Todos los derechos reservados)

Atom

Atom de GitHub es una herramienta también muy ligera y que también cuenta con multitud de extensiones para adaptarla a nuestras necesidades. Atom.

Atom (Todos los derechos reservados)

SublimeText

SublimeText al igual que los anterior esse le pueden instalar gran cantidad de extensiones, aunque es software propietario pero se puede usar para enseñanza.

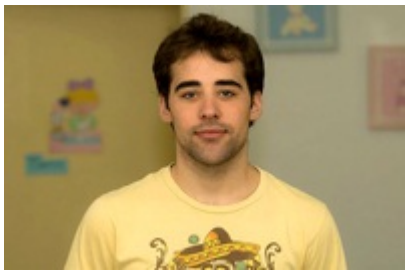
Sublime Text (Todos los derechos reservados)

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. Ésto quiere decir que puedes aprender JavaScript y crear aplicaciones con un ordenador portátil y desde cualquier lugar sin necesitar Internet para ello.

Lo que sí es importante es utilizar las extensiones que nos ofrecen muchos de los navegadores ya que te van a permitir ver las salidas por consola que anuncian errores, ver la estructura del documento, incluso en algunos casos depurarlos, etc.

4.- Integración de código Javascript con HTML.

Caso práctico



Antonio ha decidido ya el editor y los navegadores web que va a utilizar para programar con JavaScript, así que se pone manos a la obra y mira cuáles son los primeros pasos para integrar el nuevo código de JavaScript en el HTML.

Como **Antonio** ya conoce el lenguaje HTML, puesto que lo estudió en uno de los módulos que está cursando en su ciclo formativo, se centra en la parte específica de HTML que le permitirá integrar el nuevo lenguaje de programación JavaScript con el lenguaje HTML que ya conoce.

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 utilizar un fichero externo que contenga el código de JavaScript y referenciar dicho fichero. 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 código

JavaScript tendrán la extensión **.js**.

Por ejemplo:

```
<script  type="text/javascript"  src="miScript.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 **miScript.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.midominio.com/miScript.js"></script>
```

En este caso estamos referenciando nuestro script mediante una referencia absoluta, que hace referencia a nuestro dominio en primer lugar y posteriormente a la ruta dentro de nuestro domino. Es decir, estamos referenciando mediante una URL que se refiere a nuestro domino.

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

En este caso estamos referenciando nuestro script mediante una referencia relativa, que hace referencia a nuestro nuestro script desde el mismo directorio o carpeta en la que se encuentra la página **.html** en la que está incrustado dicho código, pero situado en el directorio **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. Ésto 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. Pero eso no es lo habitual hoy día.

Para insertar código Javascript en HTML (suponemos que trabajamos en HTML5) utilizaremos el siguiente código:

- Insertando el código directamente en el archivo **`index.html`**:

```
<!DOCTYPE html>
<html lang="es-ES">
    <head>
        <meta charset="UTF-8">
        <title>Hola mundo desde DWEC</title>
        <script>
            const saludar = () => {
                alert('Hola mundo desde DWEC');
            }
        </script>
    </head>
    <body>
        <h1>
            Hola Mundo desde DWEC
        </h1>
        <button onclick="saludar()">Saludar</button>
    </body>
</html>
```

- Haciendo referencia al código existente en otro archivo **`.js`**:
  Creamos un archivo nombrado **`miScript.js`** cuyo contenido es el siguiente:

```
const saludar = () => {
    alert('Hola mundo desde DWEC');
}
```

Y ahora lo referenciamos desde el archivo **`index.html`**:

```
<!DOCTYPE html>
<html lang="es-ES">
    <head>
        <meta charset="UTF-8">
        <title>Hola mundo desde DWEC</title>
        <script src="miScript.js"></script>
    </head>
    <body>
        <h1>
            Hola Mundo desde DWEC
        </h1>
        <button onclick="saludar()">Saludar</button>
    </body>
</html>
```

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 ésto 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 comprimida del script que has programado para facilitar su carga rápida.

Para que veas un ejemplo de ofuscador de JavaScript visita:

Herramienta para ofuscar JavaScript.

Lo mejor es que cambies ese paradigma 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. Lo suyo sería que lo compartieras con toda la comunidad, con la licencia que estimes oportuno, por ejemplo en GitHub.