



Diseño de interfaces web

Unidad 4: Integración de contenido interactivo: Scripts

ÍNDICE

INTRODUCCIÓN.....	3
OBJETIVOS / CAPACIDADES.....	4
PROYECTO DE LA UNIDAD.....	5
1. ELEMENTOS MULTIMEDIA E INTERACTIVOS BÁSICOS Y AVANZADOS. TECNOLOGÍAS RELACIONADAS CON LA INCLUSIÓN DE CONTENIDOS MULTIMEDIA E INTERACTIVOS.....	7
2. COMPORTAMIENTOS INTERACTIVOS. COMPORTAMIENTO DE LOS ELEMENTOS.....	10
3. HERRAMIENTAS GRÁFICAS PARA EL DESARROLLO DE CONTENIDO MULTIMEDIA INTERACTIVO. CREACIÓN DE CONTROLES Y EFECTOS VISUALES.....	12
4. MODIFICACIÓN DE LOS COMPORTAMIENTOS.....	16
Cuestionario.....	18
5. CAMBIO DE LAS PROPIEDADES DE UN ELEMENTO. AGREGAR, MODIFICAR Y ELIMINAR PROPIEDADES DE UN ELEMENTO.....	19
6. EJECUCIÓN DE SECUENCIAS DE COMANDOS.....	22
7. REPRODUCCIÓN DE SONIDO, VIDEO Y ANIMACIÓN.....	25
8. CHEQUEO DEL CONTENIDO INTERACTIVO DESDE DIFERENTES NAVEGADORES. CONFIGURACIÓN DE NAVEGADOR.....	28
Cuestionario.....	31
RESUMEN.....	32
RECURSOS PARA AMPLIAR.....	33
BIBLIOGRAFÍA.....	34
GLOSARIO.....	35

INTRODUCCIÓN

La unidad anterior trató sobre el **contenido multimedia**, algo que hace la web más **visual**. En esta ocasión, le daremos una vuelta de tuerca más a dicho contenido y veremos el concepto de **interactividad**, el cual hace referencia a cómo hacer que el usuario interactúe con nuestra web para que se sienta parte de ella.



Así pues, veremos una pequeña introducción a lo que son los **elementos interactivos**, qué **tipos** hay y **cuándo usarlos**. Para ello, veremos las herramientas que podemos utilizar y conoceremos el **lenguaje jQuery**, que nos servirá para dotar a nuestra web de **interactividad** de forma más o menos sencilla. Para acabar, veremos algunas particularidades sobre la **reproducción de audio y vídeo** en nuestras páginas web, y cómo, en ocasiones, puede sufrir **diferencias** dependiendo del **navegador o el software** que tengamos instalado.

Hay que destacar que para la mayor comprensión de esta unidad sería muy recomendable haber cursado el **módulo de Desarrollo web en entorno cliente**, donde se ven en profundidad **JavaScript** y las **librerías** relacionadas como **jQuery**.



OBJETIVOS / CAPACIDADES

En esta unidad de aprendizaje, las capacidades que más se van a trabajar son:

- ✓ Emplear herramientas y lenguajes específicos, siguiendo las especificaciones, para desarrollar componentes multimedia.



PROYECTO DE LA UNIDAD

Antes de empezar a trabajar el contenido, te presentamos la **actividad** que está relacionada con esta unidad de aprendizaje. Se trata de un **caso práctico** basado en una **situación real** con la que te puedes encontrar en tu puesto de trabajo. Con esta actividad se evaluará la puesta en práctica de los **criterios de evaluación** vinculados al resultado de aprendizaje que se trabaja en esta unidad. Para realizarla deberás hacer lo siguiente: lee el enunciado que te presentamos a continuación, dirígete al área general del módulo profesional, concretamente a la actividad de evaluación que se encuentra dentro de esta unidad, allí encontrarás todos los detalles sobre fecha y forma de entrega, objetivos... A lo largo de la unidad irás adquiriendo los conocimientos necesarios para ir elaborando este proyecto.

Enunciado:

Reproduciendo audios.

La empresa por fin te ha desvelado el motivo de las últimas actividades propuestas sobre interactividad. El próximo proyecto trata de **crear un pequeño reproductor de audio personalizado** para la página web.

La idea es que la página tenga **3 botones**:

- **Play**, para reproducir el audio.
- **Pause**, para pausar la reproducción.
- **Stop**, para parar el audio, volviendo al inicio.

Te han encargado tanto el diseño como la interactividad, así que deberás **desarrollar el código HTML, CSS y jQuery**.

Además, hay algunos otros **requisitos**:

- Los **botones** han de **cambiar de tamaño al posicionarnos sobre ellos**.
- El **archivo .mp3** a reproducir estará **incrustado en el HTML**.

- Usar algún **IDE o herramienta online** para crear el **contenido interactivo**.
- Debe **funcionar** en al menos **4 navegadores**.

1. ELEMENTOS MULTIMEDIA E INTERACTIVOS BÁSICOS Y AVANZADOS. TECNOLOGÍAS RELACIONADAS CON LA INCLUSIÓN DE CONTENIDOS MULTIMEDIA E INTERACTIVOS



Hasta el momento, durante el desarrollo del módulo nos hemos centrado sobre todo en el **aspecto visual y estético** de nuestra página web, centrándonos especialmente en las **tipografías, colores y maquetación** con las **hojas de estilo**. Además, también hemos visto las posibilidades de **añadir diferente contenido multimedia** a las páginas, haciéndolas aún más visuales y atractivas para el usuario.

Lo que veremos durante esta unidad es cómo conseguir que el **usuario se involucre** aún más en una página web a través de la interacción con distintos elementos de la página. En otros módulos has tratado con **JavaScript**, el cual permite dotar de dinamismo a una página web. Pues bien, en esta unidad trabajaremos con una **librería de JavaScript**, quizás de las más utilizadas, como es **jQuery**.



Entre otras características, **jQuery** facilita la modificación de la **estructura** del documento, dado el hecho de que permite **añadir y modificar clases CSS** y añade muchas funciones que facilitan muchas de las tareas más comunes al trabajar con la **interfaz** del sitio web.

Ahora que ya sabemos la tecnología que, mayoritariamente, vamos a utilizar, vamos a ver un poco las características esenciales que debemos conocer sobre los **elementos interactivos**:

→ Concepto.



DESTACADO

Un **elemento interactivo** es aquel que cambia cuando el usuario interactúa con él, de tal manera que le da al usuario mayor sensación de inmersión en la web.

→ Tipos de elementos interactivos.

Algunos de los **elementos interactivos más comunes** son:

- ✓ **Enlaces:** permiten al usuario la navegación entre páginas internas o externas al sitio web.
- ✓ **Cajas o áreas de texto:** permiten al usuario la introducción de texto. Por lo general las áreas de texto permiten textos más largos que las cajas.
- ✓ **Botones de opción y casillas de verificación:** permiten al usuario seleccionar opciones. Los botones son de carácter exclusivo, mientras que las casillas permiten la selección simultánea de varias opciones.
- ✓ **Listas de opciones o desplegables:** permiten al usuario la selección entre un listado, que puede ser de tipo desplegable o fijo.
- ✓ **Botones:** permiten al usuario la realización de acciones al pulsarlos.



TOMA NOTA

Todos estos elementos son comunes en los formularios, que suele ser la forma básica de interacción de un usuario con una página web.

Veamos a continuación un **ejemplo** de formulario HTML con elementos interactivos:

Ejemplo: formulario HTML con diversos elementos interactivos

Contact form

Please fill in your information and we'll be sending your order in no time.


Your Name: First Name Last Name

Your Email: e.g. hello@contact.net

Phone*: ### ### ####

Message Subject *: Other

Message *:

Verification *: ☐ I'm not a robot 



Como **elementos interactivos avanzados** principales, nos encontramos con **bibliotecas** o **conectores**, que veremos más adelante durante la unidad.

2. COMPORTAMIENTOS INTERACTIVOS. COMPORTAMIENTO DE LOS ELEMENTOS

Hemos visto ya a algunos de los elementos interactivos que nos podemos encontrar, pero ¿cómo podemos dotar a nuestra página web de un **comportamiento interactivo**?

La respuesta a esto es que existen varias **alternativas**, y, además, éstas no son excluyentes entre sí, pudiendo **usar sólo una o mezclar varias de ellas**:



- Mediante **reglas de estilo**.
- Mediante **lenguajes de programación dinámicos**.

Conozcamos un poco más a fondo estas **alternativas para el comportamiento interactivo** en la web:

→ **Reglas de estilo.**

En la unidad 2 vimos ya algunas características de las reglas de estilo que permitían dotar de algo de interactividad a la web.

En este sentido, podemos utilizar diferentes **reglas de estilo** para simular la **interacción** del usuario con la página, a través de **botones**, **enlaces** o **formularios**. Para ello usaremos las **pseudoclases**, que nos permiten entre otras cosas cambiar el **estilo de un link cuando pasamos por encima**, como podemos ver en el siguiente código.

```
a:hover {  
  color: red;  
}
```



A través de estas **pseudoclases** podemos crear elementos interactivos a través de los cuales el usuario vea cambios en la página web.

En el siguiente **link** (<https://github.com/sgvcode/gallery-css-vertical>), por ejemplo, podemos ver una **galería de imágenes creada únicamente con CSS**, sin necesidad de lenguajes de Script.

→ Lenguajes de programación dinámica.

El más conocido y utilizado, a nivel de cliente es **JavaScript**, que estudiamos en otro módulo del ciclo. Pero también podemos usar algún otro como **DHTML**.

Sin embargo, durante esta unidad nos centraremos como comentamos anteriormente en la librería más utilizada, que no es otra que **jQuery**. Es por ello por lo que recomendamos tener **conocimientos previos de JavaScript** a la hora de cursar esta unidad.



Además, para trabajar con **jQuery** también necesitaremos entender otro concepto como es el **DOM**, también estudiado en el módulo de **desarrollo web en entorno cliente**.



Cabe destacar que, gracias a **JavaScript**, podemos conseguir que nuestra página sea **realmente interactiva** y no sólo lo **simule**, como haríamos mediante las **hojas de estilo CSS**.

3. HERRAMIENTAS GRÁFICAS PARA EL DESARROLLO DE CONTENIDO MULTIMEDIA INTERACTIVO. CREACIÓN DE CONTROLES Y EFECTOS VISUALES

Dado que actualmente todo el **contenido multimedia** que se encuentra en la web es gestionado y manipulado utilizando **JavaScript**, **HTML** y **CSS**, no hay muchas herramientas que alivien cargas de nuestra tarea.

Sin embargo, centrémonos en tratar de conocer más a fondo algunas **herramientas** que nos pueden ser útiles para el desarrollo de **contenido multimedia interactivo**:

→ Herramientas de exportación de contenidos.

Existen **herramientas** que **exportan contenidos** como HTML, CSS y JavaScript. Se trata principalmente de software de pago, como **Adobe Animate CC**, que permite la **creación** de contenidos multimedia y que posteriormente los **exporta** como código HTML, CSS y JavaScript.



→ Bibliotecas.



DESTACADO

*Lo que sí se va a utilizar muy a menudo son **bibliotecas**, las cuales facilitarán enormemente la tarea ya que **automatizan las acciones más repetitivas, simplifican la sintaxis** y se encargan de **gestionar las diferencias entre navegadores**. Para ello, utilizan internamente **diferentes implementaciones** según sean o no soportadas determinadas características.*

→ Aplicaciones online.

Otro tipo de herramienta muy útil son las **aplicaciones online** que se pueden encontrar para **resolver problemas muy concretos** y que generan código automáticamente, o aquellas que nos permiten **probar código online** de manera rápida, sin necesidad de IDE's externos, como **CodePen** (<https://codepen.io/pen/>).

En este sentido, podemos encontrar, por ejemplo, los **generadores de código CSS3** que facilitan la tarea de añadir **sombras, degradados** y otros **efectos** a partir de una interfaz gráfica.

→ jQuery.

Pero como ya hemos dicho, durante esta unidad utilizaremos **jQuery**, que nos ayudará en todo este proceso. Podemos descargarlo desde su propia página web (<https://jquery.com/download/>) y, posteriormente, lo **incluiremos en nuestro archivo HTML**, dentro de la etiqueta `<head>`:

```
<script src="direccion jquery.js"> </script>
```

Una de las primeras **instrucciones** que todo código **jQuery** debe tener es la siguiente:

```
$(document).ready( function() { ... } );
```

Dentro de esta **instrucción** podremos incluir **todo el código que queramos que utilice nuestra página**.

La función más utilizada en jQuery es `$()`. Esta función se utiliza para **seleccionar** un elemento del **árbol DOM** a través del **identificador**, a través del nombre de la clase o de la etiqueta (CSS):

- ✓ `$("#miIdentificador");`
- ✓ `$("a");` : seleccionar los enlaces.
- ✓ `$("#miParrafo, a");` : seleccionar los enlaces y un identificador.
- ✓ `$(".miClase");`
- ✓ `$("a[rel]");` : con un determinado atributo.
- ✓ `$("input[@type=radio]");` : con un valor para un atributo.
- ✓ `$("/html/body/p");` : usando Xpath.

A continuación, te animamos a visualizar el siguiente **vídeo** sobre el **funcionamiento de jQuery**:

Vídeo: qué es y cómo funciona jQuery



Visualiza el siguiente vídeo sobre el funcionamiento de jQuery:
<https://www.youtube.com/embed/jd-9CoAXB4w>

Actividad de aprendizaje 1: comprender y modificar código de interactividad

Durante los últimos meses, las visitas a la página han decaído y el tiempo que pasan los usuarios visitándola es cada vez menor.

Por ello, y con el objetivo de **retener a los clientes en la página**, tu empresa está decidida a dar un giro en su forma de trabajar. Es por ello por lo que tu jefe quiere empezar a dotar de más **interactividad** a la página web.

Como justamente estáis empezando a tratar este tema, os piden vuestra ayuda. Han conseguido un ejemplo en Internet, con un par de **animaciones** para los momentos en los que el usuario pasa el **ratón sobre una imagen**.

Lo primero que tenéis que hacer es **entender y modificar el código** del ejemplo para que las imágenes se deslicen hacia la **derecha** en lugar de hacia abajo.

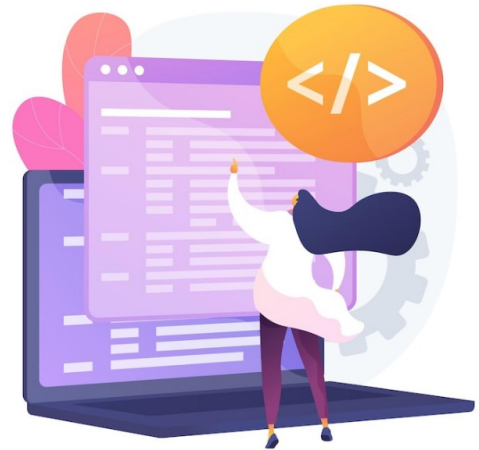
Se adjuntan los archivos [lenna.html](#), [lenna.js](#) (que habréis de modificar), [lenna.css](#) (con los estilos) y dos imágenes ([lenna.png](#) y [lenna2.jpg](#))

Trabajad en equipo con el fin de debatir y entender el código de las animaciones y modificarlas.

Entrega los **archivos necesarios** en el **espacio habilitado para ello**.

4. MODIFICACIÓN DE LOS COMPORTAMIENTOS

Ahora que ya conocemos un poco más acerca de jQuery y cómo podemos acceder a los elementos de una página web, vamos a ver algún **ejemplo** sobre cómo **modificar su comportamiento**.



Para comenzar, debemos saber que para cada **evento** disponible hay **dos posibilidades de manejo**:

- Se le puede pasar una **función** que determine su comportamiento.

```
$("p").click(function() { ... });
```

- **No se le pasa función**, por lo que se ejecuta el **evento JavaScript** del elemento.

jQuery tiene tantas **funciones** como **eventos** estándar de JavaScript. El **nombre de cada función** es el mismo que el del **evento**, pero sin el habitual prefijo "on" de los eventos. Veamos a continuación un ejemplo basado en el **evento onClick()**:

Ejemplo: evento onClick()

Por ejemplo, para el evento onClick(), usaríamos:

```
$("p").click();
```

Veamos ahora un **ejemplo de su uso**. Supongamos que tenemos una página web con un par de **cajas**:

```
<body>
  <div class="cuadro" id="primero"> Primero</div>
  <div class="cuadro" id="segundo">Segundo</div>
</body>
```

Ahora podemos incluir fácilmente **interactividad** con el usuario, de tal manera que cuando **clique** en cada una de las **cajas**, se **ejecute alguna acción definida**. Además, en el siguiente caso se muestra un **mensaje con el texto que contiene la caja seleccionada**:

```
<body>
  <script> $(document).ready(function () {
    $(".cuadro").click(function () {
      alert($(this).text());
    });
  });
</script>
</body>
```

Para ello, accedemos a los elementos con **clase "cuadro"** mediante **".cuadro"** y añadimos una función con el **código a ejecutar**, que en este caso será la alerta con el mensaje.



*Sucedería exactamente lo mismo si en lugar de usar **"cuadro"** usáramos **div**, ya que en este caso todas las **cajas** se comportarían de la misma forma.*

Llegados a este punto, te proponemos visualizar el siguiente **vídeo** para profundizar en el conocimiento de los **selectores de clase** y el uso de la **palabra reservada this**.

[Vídeo: selectores de clases y la palabra reservada this](#)



Visualiza este vídeo en el que hablamos sobre los selectores y cómo usarlos.

Cuestionario



Lee el enunciado e indica la opción correcta:

¿Cuántos botones de opción de un grupo podemos pulsar a la vez?

- a. Todos los que queramos.
- b. Sólo 1.
- c. Sólo 2.



Lee el enunciado e indica la opción correcta:

¿Cuál es la principal función de jQuery?

- a. `$[]`
- b. `${}`
- c. `$()`



Lee el enunciado e indica la opción correcta:

En jQuery, seleccionamos un identificador mediante...

- a. `#`
- b. `.`
- c. `[]`

5. CAMBIO DE LAS PROPIEDADES DE UN ELEMENTO. AGREGAR, MODIFICAR Y ELIMINAR PROPIEDADES DE UN ELEMENTO

En el punto anterior hemos visto cómo acceder a los elementos para que **reaccionen a acciones del usuario**. Veamos ahora más **ejemplos**, en este caso acerca de **cómo modificar las propiedades CSS de un elemento**.



Veamos a continuación algunos ejemplos de uso, ya sea para **cambiar propiedades CSS**, **acceder al valor de una propiedad** o **eliminarla**.

➔ Cambiar una propiedad CSS.

Supongamos que tenemos el mismo HTML del ejemplo anterior, pero ahora con **cuatro cajas**:

```
<div class="cuadro" id="primero"> Primero</div>
<div class="cuadro" id="segundo">Segundo</div>
<div class="cuadro" id="tercero">Tercero</div>
<div class="cuadro" id="cuarto">Cuarto</div>
```

Además, tenemos un **estilo CSS asociado**, ya sea **incrustado** en el propio HTML o vinculado desde una **hoja de estilos externa**.

```
div {
  width: 100px;
  height: 40;
  margin: 10px;
  padding: 10px;
  float: left;
  background-color: lightgrey;
  color: white;
}
```

La idea ahora es que, al **clickar sobre cada una de las cajas**, nos **cambie el color** de éstas.

```
$(".cuadro").click(function () {
  $(this).css('background-color', 'red')
});
```

Aquí accedemos a los **elementos de la clase “cuadro”** y cuando el usuario clicka sobre él, **cambia el color** de fondo de su caja a rojo.



DESTACADO

Para conseguir esto se usa la **palabra reservada this**, que sirve para acceder al elemento que ha provocado la acción. En este caso, cuando se **clicka una caja**, se **cambia el color** de esa caja únicamente.

Si en lugar del **this** usáramos también **“#cuadro”** a la hora de cambiar la propiedad CSS, esta vez estaríamos cambiando el color de los 4 cuadros a la vez.

➔ **Acceder al valor de una propiedad CSS.**

A parte de para cambiar una propiedad CSS, también podemos utilizar jQuery para **acceder al valor** de éstas. Por ejemplo, de esta manera podemos conseguir mediante el siguiente código que, con cada clic del usuario, la caja alterne entre color verde y rojo.

```
$(".cuadro").click(function () {  
    if ($(this).css('background-color') != "rgb(255, 0, 0)") {  
        $(this).css('background-color', 'red')  
    } else {  
        $(this).css('background-color', 'green')  
    }  
});
```

Mediante el **formato RGB** comprobamos que el color de fondo de la caja no sea rojo y, si no es así, la ponemos en dicho color. Si comprobamos que estaba en rojo, la cambiamos a verde, lo cual produce la **alternancia de colores**. De esta manera podemos **modificar propiedades ya existentes** de los elementos, o crear nuevas.

➔ Eliminar una propiedad CSS.

Si lo que queremos es **eliminar una propiedad**, bastaría con asignarle un **valor vacío** a dicha propiedad. Por ejemplo, podríamos hacerlo de la siguiente manera:

```
$(this).css('background-color', '')
```

Así conseguimos que el elemento **deje de tener valor** para la propiedad de color de fondo.

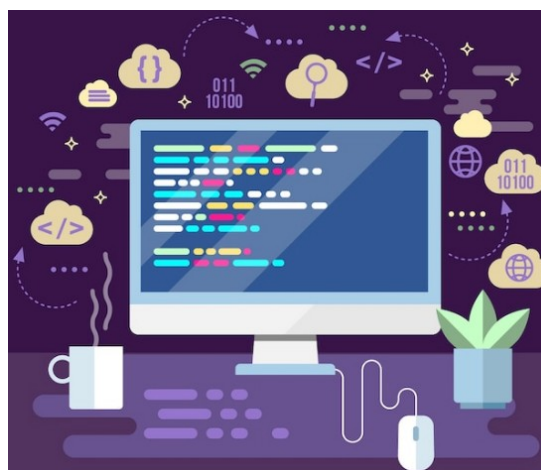
6. EJECUCIÓN DE SECUENCIAS DE COMANDOS



DESTACADO

A la hora de trabajar con **jQuery**, como con cualquier otro lenguaje de programación, podemos ejecutar diferentes instrucciones unas detrás de otras, o saltar entre ellas mediante **sentencias condicionales o iterativas**.

En este punto vamos a ver algunas **características** interesantes acerca de la ejecución de **secuencias de comandos**, entendiendo éstas como el hecho de **ejecutar varios comandos uno detrás de otro**. Para ello, debemos introducir dos conceptos de jQuery que nos serán de gran utilidad: los **métodos fadeOut y fadeIn** y **callback**.



→ Métodos fadeOut y fadeIn.

Imaginemos que queremos **cambiar el color** de una caja, como en los ejemplos anteriores, pero esta vez no queremos hacerlo de forma brusca, si no que queremos darle un efecto de **desvanecimiento** al cambio de color. Para eso podríamos utilizar los **métodos fadeOut** y **fadeIn** de jQuery, de tal manera que quedaría así:

```
$(this).fadeOut(100)
$(this).css('background-color', 'red')
$(this).fadeIn(100);
```

El problema es que, si ejecutamos este código, veremos que **no se produce el efecto deseado**, ya que el cambio de color se produce antes de que finalice el primer fadeOut. La idea es que, en primer lugar, el botón desaparezca lentamente, para que luego se cambie el color sin que lo veamos y vuelva a aparecer.

Aprovechando que jQuery permite **concatenar comando**, podríamos hacer algo parecido a esto:

```
$(this).fadeOut(100).css('background-color', 'red').fadeIn(100);
```

Esto **acorta el código** al asociar varias acciones consecutivas a un mismo elemento. En este caso el resultado sigue siendo el mismo, por lo que aun no es el que esperamos realmente.

→ **Callback.**

Para solucionar este problema vamos a introducir un nuevo concepto, común en muchos lenguajes de programación y especialmente en los más modernos. Hablamos de los **callback**.



DESTACADO

*Podemos definir a los **callback** de jQuery como una herramienta que nos permite ejecutar funciones una vez ha **terminado el efecto actual**.*

Esto es precisamente lo que queremos en nuestro caso, así que veamos cómo conseguirlo:

```
$(this).fadeOut(100, function(){
    $(this).css('background-color', 'red').fadeIn(100);
});
```

La clave aquí está en que en el mismo efecto de **fadeOut** le especificamos la función o **callback** que queramos que se ejecute al finalizarlo. Además, dentro del propio callback también nos aprovechamos del método de **concatenar acciones** visto antes.

De esta manera sí que conseguiríamos que el botón, antes de cambiar de color, **desaparezca por completo de manera progresiva** para luego volver a aparecer.

Para finalizar este apartado, te proponemos visualizar el siguiente **vídeo** sobre la herramienta **callback**.

Vídeo: herramienta callback



Visualiza el siguiente vídeo sobre la herramienta callback y su uso.

7. REPRODUCCIÓN DE SONIDO, VIDEO Y ANIMACIÓN

Antes de la llegada de HTML5 y las mejoras aportadas por CSS3, cuando se querían añadir **elementos interactivos** a un sitio web debía recurrirse a tecnologías ajenas al navegador como los **applets de Java**, o mucho más frecuentemente a **Flash**.



*Estas tecnologías hoy en día están obsoletas y han sido reemplazadas por el uso de **HTML5**, **CSS** y **JavaScript**.*

*Esto **simplifica** mucho la configuración de los navegadores, ya que no es necesario descargar ningún **conector especial** (como requería Flash) ni la **máquina virtual** de Java. Al ser tecnologías nativas del navegador, pueden utilizarse directamente.*

Aunque el desarrollador no tenga que preocuparse de que el usuario tenga que instalar ningún conector en particular, sí que hay dos opciones que se pueden modificar o desactivar que pueden hacer que **el sitio web deje de funcionar y mostrarse correctamente**:



- ➔ **Desactivar el uso de cookies**: dado que no se podrá recordar la información del usuario, algunos sitios web serán directamente inaccesibles como, por ejemplo, los sitios web de **comercio electrónico**.
- ➔ **Desactivar el uso de JavaScript**: en algunos casos el sitio será **inservible**. Por ejemplo, una web que incluya mapas de Google Maps no puede funcionar sin JavaScript; en cambio, otros sólo lo utilizan para **añadir algunos efectos** y para éstos sí puede presentarse una solución alternativa.

Estos casos deben tenerse en cuenta y, como mínimo, se debe **mostrarse un mensaje avisando al usuario** de que el sitio no puede funcionar o tiene limitadas

sus funcionalidades dado que estas opciones están desactivadas o no están disponibles.

Veamos a continuación un **ejemplo de uso de métodos alternativos** para reproducir **audio y vídeo** cuando el navegador **no soporta el formato HTML5**.

Ejemplo: uso de métodos alternativos para reproducir audio y vídeo

Cabe destacar que podemos recurrir a **métodos alternativos** para reproducir **audio** y **vídeo** cuando el navegador no soporta **HTML5**. Actualmente no existe ninguna solución soportada universalmente, así que lo único que se puede hacer es **mostrar un mensaje al usuario**. Veamos un ejemplo en código:

```
<video>
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  Lo sentimos, pero tu navegador no soporta vídeo de HTML5
</video>
```

En cuanto a la **reproducción de audio**, podemos encontrarnos con el siguiente caso:

- En primer lugar, se intenta utilizar como fuente del vídeo un archivo en **formato MP4**.
- Si este formato no es reconocido, se intenta con el **formato WEBM**.
- Si ninguno de los dos es reconocido, se muestra un **mensaje informando al usuario**.



*Sea cual sea el modo que empleemos para incluir sonido, video y/o animaciones en una página web, los archivos deben estar en un **formato apropiado para la web**, teniendo el **menor tamaño posible** sin que afecte a la calidad necesaria para que el usuario pueda oír y/o ver su contenido correctamente.*

Actividad de aprendizaje 2: creación de animaciones para botones

Ahora que ya entendéis cómo funcionan las **animaciones interactivas** y cómo podemos **modificar su comportamiento** mediante el uso de jQuery, la empresa quiere que empecéis a desarrollar la **primera animación** que usarán en la página web.

Concretamente, necesitan que creéis una pequeña página web con **3 botones a modo de menú**. Mediante **estilos CSS**, estos botones tendrán **color verde**.

La idea es que, usando **jQuery**, cuando el usuario pase el ratón por encima de los botones, se pongan de un color diferente cada uno.

Se trata de **trabajar por parejas** en la elaboración del código jQuery.

Entrega los **archivos** necesarios en el **espacio habilitado para ello**.

8. CHEQUEO DEL CONTENIDO INTERACTIVO DESDE DIFERENTES NAVEGADORES. CONFIGURACIÓN DE NAVEGADOR

Hemos visto en los diferentes módulos del ciclo que una de las cosas más importantes de un desarrollador es asegurarse de que sus proyectos van a **funcionar correctamente**.



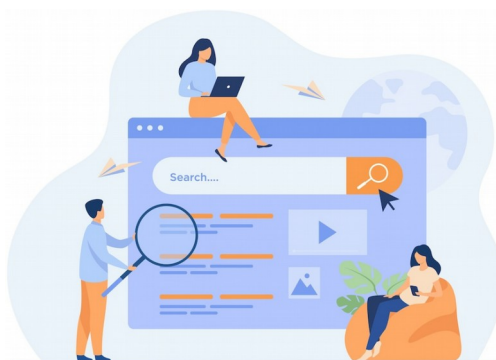
DESTACADO

*En el caso de la **programación web** esto implica que no sólo tienen que **funcionar correctamente**, sino que lo tiene que hacer para **todos los navegadores** y, además, en la medida de lo posible, la aplicación debería **verse lo más parecida posible en todos ellos**.*

Es por ello por lo que se hace necesario que el desarrollador **pruebe** el sitio en la **mayoría de navegadores disponibles**, ya que puede variar mucho su funcionamiento, especialmente en cuanto a **contenido multimedia** se refiere.

→ Configuración del navegador.

La **configuración del navegador** también es un factor que puede influir en el funcionamiento, como hemos visto en el punto anterior, dado que la **desactivación de las cookies o de JavaScript** puede arruinar por completo la experiencia del usuario.



Para intentar solucionar estos problemas, una de las primeras acciones que se pueden acometer es intentar separar lo máximo posible la **estructura de la web** del **diseño** y su **funcionalidad**, como dicta el **MVC**.



Podemos hacer que todo nuestro **código JavaScript** o **jQuery** se encuentre en un **fichero aparte**. Así, al menos el usuario que tenga JavaScript desactivado podrá seguir disfrutando de la página, aunque con algunas **limitaciones**.

→ Paradigmas de trabajo.

Estos tipos de **situaciones** se pueden encontrar siempre que se trabaje con **HTML5, CSS3 y JavaScript**. Pueden afrontarse desde **dos ángulos distintos**:

- ✓ **Progressive enhancement**: se empieza a desarrollar el sitio con las **funcionalidades básicas** que no requieren ninguna tecnología y se va mejorando la experiencia de usuario **añadiendo** (y comprobando) **nuevos comportamientos** paso a paso.
- ✓ **Graceful degradation**: con esta técnica se **trabaja de manera inversa**, primero se desarrolla el sitio con todas las tecnologías que se necesitan y después se añaden **alternativas de visualización/ejecución** para las partes que no estén disponibles globalmente.



En cualquiera de las dos vertientes sigue siendo muy importante la **separación del código dinámico del resto de la página**.

Además, nos podemos aprovechar del hecho de que las hojas de estilo CSS se aplican en **cascada**, lo cual nos permite que una declaración pueda ser **sobrescrita** por las posteriores.

Así pues, podríamos hacer que las **reglas para los navegadores más antiguos estuvieran en primer lugar**, de tal manera que los más modernos podrían **reemplazarlas** por las que se encuentren después en la hoja de estilo.

Para afianzar conocimientos, te proponemos visualizar el siguiente **vídeo** sobre cómo **desactivar JavaScript en los navegadores de Safari, Chrome y Firefox**.

Vídeo: desactivar JavaScript en navegadores



Visualiza el siguiente vídeo sobre la desactivación de JavaScript en navegadores.

<https://www.youtube.com/embed/dttMOdm6knk>

Actividad de aprendizaje 3: informe de diferencias de comportamiento

Tu jefe está encantado con el trabajo realizado en las actividades anteriores, sin embargo, se ha dado cuenta de un pequeño detalle. Cuando ejecuta las páginas en **distintos navegadores**, éstas tienen pequeñas **diferencias de comportamiento**. Por eso, para evitar este hecho en un futuro, quiere estar seguro del **por qué**. Así que te pide:

- Visualizar las siguientes animaciones hechas con **HTML5, CSS3 y Javascript** en los **distintos navegadores** que tienes instalados en tu ordenador.
 - <http://bomomo.com/>
 - <http://tumult.com/hype/gallery/BirthdayParty/BirthdayParty.html>
 - <http://tumult.com/hype/gallery/WooGrit2/WooGrit2.html>
 - <http://peterned.home.xs4all.nl/3d/>
- ¿Se pueden ver en todos los navegadores directamente y de la misma forma?
- ¿Alguno necesita utilizar el plugin de Flash, ya que no soporta HTML5?

Entrega el documento explicativo en el espacio habilitado para ello.

Cuestionario



Lee el enunciado e indica la opción correcta:

¿Qué hace el siguiente código?

```
$(".cuadro").click(function(){$(this).css('background-color','red')}});
```

- a. Cambia el color al pasar por encima.
- b. Cambia el color de un elemento al clicar sobre él.
- c. Cambia el color de varios elementos de tipo "cuadro" al pulsar sobre uno de ellos.



Lee el enunciado e indica la opción correcta:

¿Cómo podemos eliminar el valor de un atributo CSS en jQuery?

- a. nil.
- b. 'nil'.
- c. ''.



Lee el enunciado e indica la opción correcta:

¿En una hoja CSS con varios estilos iguales cuál se aplica?

- a. El último que se encuentre.
- b. El primero que se encuentre.
- c. Ninguno de ellos.

RESUMEN

Al finalizar esta unidad ya sabemos lo que es el **contenido interactivo** de una página web, **qué tipos hay** y **cómo podemos utilizarlos**. Para ello, hemos partido de la idea de que el alumno tiene algunos **conocimientos previos** de **lenguajes de script**, concretamente de **JavaScript**, lo que le hará mucho más fácil el uso de **jQuery**.

Como hemos visto, **jQuery** es la **librería** más utilizada para este tipo de propósitos y permiten realizar contenido interactivo de forma sencilla. Por ello hemos visto cosas como, por ejemplo, cómo podemos **cambiar las propiedades** de un objeto mediante **jQuery**. Además, podemos incluso **cambiar sus propiedades CSS**.

Además, mediante el uso de jQuery podemos **crear nuevos elementos en nuestra página web** y hacer que el usuario pueda **reaccionar** a algunos ya existentes. Finalmente, hemos conocido el concepto de **callbacks**, el cual permite la ejecución de ciertos comandos una vez que ha acabado otro, permitiendo así una especie de **sincronía en el código**.

Como ya dijimos en la introducción, todos los conceptos vistos en esta unidad se entienden mejor junto a otros módulos del ciclo, especialmente los que tratan sobre **lenguajes de script**.

RECURSOS PARA AMPLIAR



PÁGINAS WEB

- CSS play doing it with style: <http://www.cssplay.co.uk/menu/gallery.html> [Consulta septiembre 2022].
- jQuery Callback Function:
https://www.w3schools.com/jquery/jquery_callback.asp [Consulta septiembre 2022].



BIBLIOGRAFÍA



PÁGINAS WEB

- CSS | MDN: <https://developer.mozilla.org/es/docs/Web/CSS> [Consulta septiembre 2022].
- HTML: lenguaje de etiquetas de hipertexto | MDN: <https://developer.mozilla.org/es/docs/Web/HTML> [Consulta septiembre 2022].
- W3C: <https://www.w3.org/> [Consulta septiembre 2022].



GLOSARIO

- **DHTML (HTML Dinámico)**: usado en páginas interactivas sin necesidad de código de script.
- **DOM**: el modelo de objeto de documento es una representación completamente orientada al objeto de la página web y puede ser modificado con un lenguaje de script.
- **MVC (Modelo Vista Controlador)**: paradigma de programación que separa los datos del diseño y la funcionalidad.
- **RGB (Red Green Blue)**: formato de colores que combina los canales: rojo, verde y azul.

