Contenidos web interactivos.

Caso práctico

Como todas las semanas, se reúnen los miembros del equipo de trabajo de la empresa **BK programación** para comentar las incidencias de la semana anterior.

Esta semana tienen algo que celebrar ya que el proyecto de la panadería "Migas Amigas" está prácticamente listo y si lo entregan antes de la fecha límite, que es dentro de dos semanas, tendrán una bonificación ya que el gerente de la panadería quería que estuviera lista para la campaña navideña.

–¿Habéis verificado bien el código fuente con las herramientas de la W3C?–pregunta Ada a todos los presentes.

–Sí –contesta Antonio que continúa diciendo: –lo hemos validado para el XHTML 1.0 y las CSS3, y ya hemos añadido los iconos en el pie de página.

–Y ¿habéis verificado la accesibilidad?–vuelve a preguntar Ada.

–¡Pues claro! –le responde María, que continúa diciendo: –Hemos empleado las herramientas de verificación automática y –es Juan el que termina la frase– hemos revisado manualmente todos los criterios de éxito para que estuviera conforme con el nivel A. Estoy seguro de que conseguirá la subvención de su ayuntamiento.

Ada interviene preguntando: –¿Se la habéis dejado utilizar a alguna persona para ver si le resultaba fácil de usar?

–La verdad es que todos se la hemos enseñado a alguien para que nos diera su opinión. Yo en concreto se la enseñé a mi abuela que aunque todavía es joven no suele utilizar Internet muy a menudo y no tuvo ningún problema. Hasta me ha dicho que la avise cuando esté funcionando para poder escribirles con el formulario de contacto y encargar el roscón de Reyes –responde Ana.

–Habéis hecho un trabajo estupendo –les dice Ada y añade –y aún nos quedan dos semanas de plazo. ¿Creéis que se podría mejorar en estas dos semanas algo?

–Yo creo que sí –contesta Juan, que continúa diciendo: –Podríamos hacer la página más interactiva de modo que la experiencia del usuario sea más agradable. Aunque el formulario es accesible con el ratón y con el teclado y está preparado para las ayudas técnicas de las personas con discapacidad, podríamos controlar los eventos del usuario para que cuando éste pase por encima de algún enlace o algún elemento del formulario, o cuando un elemento se active, haya una cambio más evidente del que hay ahora. Y lo podemos hacer complementado la hoja de estilos y añadiendo algo de código con Javascript para controlar los eventos de los elementos. Sí, ya sé, no me mires de esa manera –Juan observa que Ada lo está mirando – ya se que hay que separar el contenido del aspecto y del comportamiento. No te preocupes, el script se puede vincular al documento igual que las hojas de estilo y después validaremos todo de nuevo.

–¿Qué pasa si un usuario no tiene activado el Javascript? –pregunta Carlos.

–Si se hace todo como dice Juan el único problema que tendrá ese usuario es que se perderá esa experiencia, porque el resto de la funcionalidad de la página permanecerá intacta ¿No es así Juan? –responde Ada, y como todas las semanas da por terminada la reunión diciendo: –Pues a trabajar todos.

1.- Elementos interactivos.

Caso práctico

Juan es un experto en desarrollo de aplicaciones informáticas y aplicaciones web. Conoce, al igual que Ada, muchos lenguajes de programación, tanto los que se utilizan del lado del cliente como los del lado del servidor.

Hasta ahora ha utilizado el lenguaje Javascript sin preocuparse demasiado de la accesibilidad y no tenía en cuenta si el usuario podía tener deshabilitado de funcionamiento de Javascript y podía perder parte de la funcionalidad de la página.

Ahora tiene mucha más experiencia y su labor en el equipo consiste en lograr que todos los compañeros y compañeras del equipo de BK programación realicen sus tareas teniendo en cuenta que deben hacer la vida más fácil y agradable al usuario.

Hemos visto en unidades anteriores algunos recursos gráficos: imágenes, sonido, vídeo y animaciones como complementos importantes del contenido de una página web y que toda persona que se dedica al diseño de interfaces web debe conocer.

En esta unidad trataremos el empleo de los elementos interactivos en una web. El empleo de los elementos interactivos no es llenar la página web con recursos gráficos en movimiento, sino que, es una forma de acercarse al usuario y hacerlo participar, intentando lograr una web más dinámica mediante el establecimiento de un canal de comunicación con el usuario. Si los usuarios perciben la sensación de que hay alguien ahí detrás que está pendiente de ellos, que le contesta a sus dudas, que le permite opinar, sin duda volverá.

Es importante tener en cuenta que, al utilizar los elementos interactivos, no debemos olvidar el compromiso que tenemos con la **accesibilidad** de todos usuarios a la web y con la **usabilidad** del sitio. Recuerda siempre que no podemos sacrificar aspectos tan importantes como la accesibilidad y la usabilidad en aras de una mayor interactividad.

El diseño de un sistema interactivo debe estar centrado en el usuario y debe permitir a cualquier persona percibir el proceso interactivo como una experiencia agradable y con el que pueda obtener los resultados esperados.

Dotar de interactividad a una página siempre debe pensarse como algo adicional a conseguir después de que ya se ha conseguido todo lo imprescindible: estandarización, accesibilidad y usabilidad.

1.1.- Elementos básicos y avanzados.

El empleo de elementos interactivos en la web, nos van a servir para tratar de fidelizar a nuestros usuarios. Hay muchos elementos que permiten interactuar con el usuario: una zona de "Deja tu comentario", "Mándanos tu opinión", etcétera.

> Un elemento interactivo es aquel que cambia cuando el usuario interactúa con él.

Resumen textual alternativo.

Los foros, blogs, encuestas, comentarios, etcétera son elementos interactivos que puede o no tener un sitio web porque permiten la comunicación con el usuario y también entre los usuarios pero, por la definición de elemento interactivo dada en el párrafo anterior podríamos considerar también como elementos interactivos los siguientes:

- **Los enlaces**. Todos los elementos ancla del HTML , son elementos interactivos, puesto que, aunque no se definan estilos para ellos, el color que muestran al principio, cuando se hace clic sobre ellos y después de ser visitados, es diferente. Además una vez que se pulsa un enlace, debería haber algún cambio en la página.
- Todos los **objetos propios de los formularios** son elementos interactivos:
  - La cajas de texto son los elementos donde el usuario puede escribir alguna información compuesta por un número de caracteres no muy extenso. Al principio pueden contener un texto por defecto o estar vacías y, mediante la interacción del usuario que escribe, borra, inserta o pega texto en ellas, cambian su contenido.
  - Los botones de opción son elementos donde el usuario puede elegir una opción entre varias. Con los botones de opción, también se puede dejar uno de ellos preseleccionado por defecto o bien ninguno y, es el usuario el que decide cuál seleccionar. El aspecto de los botones de opción cuando están seleccionados es diferente de cuando no lo están.
  - Las casillas de verificación son elementos que el usuario puede elegir seleccionar o no. Las casillas de verificación no se suelen presentar preseleccionadas por defecto pero cuando se seleccionan cambian su aspecto.
  - Las áreas de texto son parecidas a las cajas de texto pero permiten escribir varias líneas de texto. Se suelen emplear para el envío de comentarios. Pueden, al igual que las cajas, mostrar un texto por defecto que guíe al usuario sobre lo que tiene que hacer. En la imagen que ilustra este apartado se ve un área de texto que contiene el texto "Escriba aquí sus temas de interés".
  - Las listas de opciones es un elemento interactivo en el que el usuario puede elegir uno o varios elementos de la lista, según cómo sea su configuración, con la misma funcionalidad que los botones de opción, si sólo permite elegir una opción, o con la misma funcionalidad que las casillas de verificación, si permite seleccionar varias opciones de la lista, aunque éste uso no es el más común. La diferencia está en que ocupa menos espacio en el formulario. Se suele emplear para que el usuario elija su nacionalidad, el día, el mes o el año de su nacimiento.
  - Los botones son los elementos que permiten al usuario confirmar una determinada acción. Su aspecto suele cambiar cuando se pulsa de forma que el usuario identifica dicha pulsación. Aunque en la imagen no se ve ningún botón, todos los formularios suelen disponer de dos botones: uno que es el encargado de ejecutar la acción cuyo nombre dependerá de la acción a ejecutar y otro, que es el encargado de limpiar los datos y dejar el formulario vacío de nuevo.

Los enlaces y objetos propios de formularios son los elementos interactivos básicos necesarios para crear los foros, blogs, etcétera.

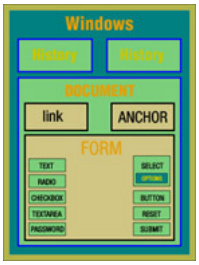
1.2.- Comportamientos interactivos.

Para añadir un comportamiento interactivo a los elementos ya mencionados y que el usuario tenga la sensación de tener el control de la página podemos hacer uso, tal y como dijimos en la unidad anterior de:

- Las reglas de estilo.
- Los lenguajes de programación dinámicos.

Podemos utilizar las reglas de estilo para simular la interacción con los botones, enlaces, elementos de formulario, etcétera. Para ello nos servimos de las pseudoclases link, visited, hover, active, focus que ya vimos la unidad de hojas de estilo.

También podemos usar el HTML Dinámico (DHTML), del que la Wikipedia dice que "designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript), el lenguaje de hojas de estilo en cascada (CSS) y la jerarquía de objetos de un DOM".



Que el **HTML** sea **estático** significa que no está generado por ningún programa ejecutado en el lado del servidor escrito en un lenguaje de programación como podría ser PHP, ASP.NET o Perl, sino que se sirve el documento HTML tal cual está escrito. Todos los usuarios reciben el mismo documento.

JavaScript es un lenguaje de guiones que sirve para extender las capacidades del lenguaje HTML. Su sintaxis es muy simple. Es un lenguaje interpretado que se ejecuta en la máquina del usuario pudiendo éste ver su código cuando consulta el código fuente de la página. El código debe ser descargado antes de poder ser ejecutado por lo que suele ir escrito en la cabecera del documento, aunque también se pueden poner scripts en el cuerpo del documento.

El **modelo de objetos de documento** (DOM) define la forma en la que se relacionan entre sí los objetos y elementos que forman parte de un documento, y su relación con el navegador. Los objetos del modelo tienen en el navegador una relación descendente entre sí. Esta relación es la que emplea Javascript para reconocer a cada objeto de una página de forma individual.

El objeto de nivel superior en el DOM es el Navegador en sí (Browser). El objeto del siguiente nivel en el DOM es la ventana del navegador (Window) y el siguiente son los propios documentos visualizados en el navegador (Document). La imagen que ilustra este apartado muestra mediante cajas los objetos del modelo y la relación jerárquica que existe entre ellos. Las cajas más externas representan un nivel superior y las más internas un nivel inferior.

En esta unidad nos centraremos en la interacción de objetos con **jQuery**, usando HTML5 y CSS3. jQuery es una biblioteca de JavaScript, que permite simplificar la forma de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. jQuery es un software libre y de código abierto.

jQuery es un lenguaje muy extenso, pero en esta unidad nos centraremos en lo más relacionado con el diseño de interfaces de usuario y su interactividad, aunque es inevitable empezar con los conceptos básicos del lenguaje.

Autoevaluación

El título incluido dentro de una etiqueta del HTML <h1> ¿es un elemento interactivo?

- ○ Verdadero.
- ○ Falso.

No es correcto. Un título no es un elemento interactivo. Sería correcta ese título hiciera a la vez de enlace.

Muy bien. La etiqueta <h1> se suele utilizar para destacar un título mediante una letra de gran tamaño.

Solución

1. Incorrecto
2. Opción correcta

Debes conocer

Desde el siguiente enlaces accederás a la página de Saúl González Fernández donde se dan las nociones de la sintaxis básica de Javascript.

Sintaxis básica de Javascript.

En el siguiente enlace conocedermos más sobre el modelo de objetos de documento (DOM) y como acceder a el.

Introducción al DOM.

1.2.1.- Introducción a jQuery.

¿Cómo incorporar jQuery a una página web?

La librería jQuery es solo un archivo JavaScript, y para utilizarla, es necesario descargarla e incluirla en los documentos web que se desee utilizar. La sentencia que se utiliza para hacer referencia a esta librería es la siguiente:

```
<head>
          <script src=”jquery.js”>
          </script>
</head>
```

Si no se desea descargar y alojar jQuery, puede ser incluirla desde un CDN (Content Delivery Network). Así, por ejemplo podría ser cargada directamente desde el CDN que mantiene Google:

```
<head>
       <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"> </script>
</head>
```

Desde el CDN que mantiene Microsoft:

```
        <head>
              <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.js">  </script>
        </head>
```

o desde el del propio jQuery

```
<head>
      <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.js">  </script>
</head>
```

Debes conocer

Toda la información oficial sobre jQuery está disponible en el sitio http://jquery.com

En el siguiente enlace podemos encontrar la descarga de las diferentes versiones de jquery.

¿Cuál es la sintaxis de jQuery?

La sintaxis de jQuery es la siguiente: **$ (selector). acción ()**

- Con el símbolo $ se define el acceso jQuery
- Entre paréntesis se especifica el selector, que permitirá seleccionar los elementos HTML.
- A continuación se especifica a realizar sobre el/los elemento/s seleccionado/s.

Algunos ejemplos sencillos que se pueden ver para comprender la sintaxis serían los siguientes:

- $ ("h1").hide() - oculta todos los elementos <h1>.
- $ (".mensaje").hide() - oculta todos los elementos con class="mensaje ".
- $ ("#test").hide() - oculta el elemento con id = "test".

En el siguiente ejemplo se muestra el código completo de un documento que contiene dos párrafos y un botón. Mediante el código jQuery la interactividad introducida permite que cuando se haga click sobre el botón se oculten todos los elementos <p>.

```
<!DOCTYPE html>

<html lang="es">

<head>

     <script  src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.0/jquery.min.js"></script>

     <meta charset="iso-8859-1">

      <script>
```

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

                $("#btn").click(function(){

                    $("p").hide();

                });

            });

        </script>

    </head>

    <body>

        <p id="pa"> este es el primer párrafo </p>

        <p id="pb"> este es el segundo párrafo </p>

        <button id="btn">Click me</button>

    </body>

    </html>
```

The Document Ready Event

Como se verá a lo largo de esta unidad, todos los métodos de jQuery en los ejemplos se encuentran dentro del evento ready(), pues es el código que se propone para ejecutar las instrucciones una vez que ha cargado la página:

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

  ...

});
```

El objetivo de este código es evitar que se ejecute código antes de que el documento haya terminado de cargarse. Así por ejemplo, algunos casos en los que pueden dar fallos son los siguientes:

- Tratar de obtener el tamaño de una imagen que no se ha cargado todavía.
- Tratar de ocultar un elemento que no se ha creado todavía.

También existe un método más corto para el evento ready():

```
ready():

$(function(){

 // métodos jQuery


});
```

1.2.2.- Selectores.

Por un selector se entiende en jQuery lo mismo que en CSS: una forma de seleccionar o elegir uno o más elementos de nuestro documento HTML. Para luego poder aplicar funcionalidad sobre estos elementos.

Los selectores de jQuery permiten seleccionar elementos HTML en función de su id, clase, tipo, atributos, valores de atributos, etc. Todos ellos comienzan con el signo $ y paréntesis.

- **Selectores básicos:**
  - **Selector universal:** selecciona todos los elementos. Sintaxis $("*")
  - **Selector de elemento:** selecciona elementos en función de sus nombres de etiqueta. Así por ejemplo, el código $("h1"), selecciona todos los elementos <h1> de la página.
  - **El selector #id:** selecciona el elemento con dicho id. Así por ejemplo, el código $("#test"), selecciona todos los elementos con id=test.
  - **El selector .class:** selecciona una clase específica. Así por ejemplo, el código $(".mensaje") selecciona todos los elementos con class=mensaje.
  - **Grupos de selectores:** permite agrupar distintos selectores separándolos por comas.
- **Selectores de atributos:** Permiten seleccionar elementos con ciertos atributos o con determinados valores en los mismos. Así por ejemplo, el código**:**
  - $("a[rel]") selecciona aquellos elementos <a> que tengan el atributo “rel”.
  - $("a[href='http://www.google.es']") selecciona todos los elementos <a> con atributo href = <a href="https://educacionadistancia.juntadeandalucia.es/formacionprofesional/pluginfile.php/19441/mod_scorm/content/0/#">http://www.goc
  - $ ("a[href^='http://www.google.es/']") selecciona todos los elementos <a> en los que el valor del atributo href empieza por la cadena indicada.
- **Algunos selectores de widgets:**
  - **Pseudo-clase button**, selecciona todos los elementos tipo button: $(":button")
  - **Pseudo-clase checkbox**, selecciona todos los elementos tipo checkbox: $(":checkbox")
  - **Pseudo-clase file**, selecciona todos los widget de tipo archivo: $(":file")
  - **Pseudo-clase header**, selecciona todas las cabeceras: $(":header")
  - **Pseudo-clase imagen**, selecciona todas las imágenes: $(":image")
  - **Pseudo-clase input**, selecciona todos los widgets de tipo input: $(":input")
  - **Pseudo-clase contraseña**, selecciona todos los elementos password: $(":password")
  - **Pseudo-clase activo**, selecciona todos los elementos que estén activos $("input:enabled")
  - **Pseudo clase inactivo**, selecciona todos los elementos que no estén activos $("input:disabled")
  - Etc.
- **Selector de descendientes:** selecciona elementos que desciendan de otro elemento.
  - **Selector de hijos:** selecciona elementos que sean hijos directos de otro elemento Así por ejemplo, la sentencia $("li > p") selecciona todos elementos <p> que sean hijos directos de <li>.
  - **Pseudo clase hijo:** selecciona el enésimo hijo de un elemento $("tr:nth-child(n)"), donde **n** representa la posición del mismo.
  - **Pseudo clase primer hijo:** selecciona el primer hijo de un elemento $("tr:first-child")
  - **Pseudo clase último hijo:** selecciona el último hijo de un elemento $("tr:last-child")
  - **Pseudo clase hijo único:** selecciona los elementos que sean hijos únicos de otros elementos $("div:only-child")
  - **Pseudo clase primero:** selecciona el primer elemento de un grupo de elementos. $("td:first")
  - **Pseudo clase último:** selecciona el último elemento de un grupo de elementos $("td:last")
  - Etc.
- **Otros selectores.**
  - **find:** permite seleccionar un conjunto de elementos descendientes de un conjunto de elementos previamente seleccionados.

```
/* Selecciona la tercera imagen (empezando por 0) de las imagenes que pertenezca a la clase imágenes (.imagenes). A dicha imagen le cambiar el b
 $($(".imagenes").find("img")[2]).css("border","red 10px solid");
```

- **firts, last:** selecciona el primer o último componente de un conjunto de elementos.
- **each:** permite realizar una iteración por un conjunto de elementos seleccionados. En cada iteración el elemento seleccionado podemos referenciarlo mediante $(this).

```
/* Seleccionamos todos los elementos que tienen clase miniaturas y obtenemos el origen y la posicion que ocupan. Componemos una cadena que poste
/* mediante $(this) hacemos referencia al objeto de cada iteración.*/
/* index nos proporciona el número de iteración o elemento empezando por cero*/

 $(".miniaturas").each(function(index){
                    txt=txt+"Imagen: "+parseInt(index+1)+" tiene como origen: "+$(this).attr("src")+"   "+"</br>";
                });

 /*Creamos una nueva etiqueta al final de nuestro documento e insertamos en ella los orígenes de las imágenes*/
                $("body").append("<p>"+txt+"</p>");
```

- **eq:** selecciona un elemento de un conjunto mediante un índice. Se comienza a contar desde 0, así el índice 0, se refiere al primer elemento.

```
/* Selecciona el elemento perteneciente a la clase miniaturas que ocupa la posición 1 empezando por 0 */<br />/* Si tenemos tres elementos selec
/* En caso de tener valores negativos empezaríamos por el final */
$(".miniaturas").eq(1).css( "background-color", "red" );
```

- **slice:** similar a eq(), pero podemos indicar un conjunto de elementos, indicando un elemento de inicio y uno de fin.

```
/* Selecciona los elementos pertenecientes a la clase miniaturas que se encuentren entre la posición 2 y 4, el 4 indica el fin, con lo que no se
/* Si tenemos seis elementos  seleccionamos los elementos  3 y 4, ya que empezaríamos a contar desde 0 --> 0 1 2 3 4 5 6 */
/* En caso de tener valores negativos empezaríamos por el final */
<br />$(".miniaturas").slice(2,4).css("border","3px solid blue");
```

Recomendación

Como material complementario para reforzar el concepto de selector, así como los distintos tipos de selectores y algunos ejemplos, se recomiendan consultar los siguientes sitios:

jQuery Selectors(w3schools)

Selectors (jQuery API)

1.2.3.- Eventos.

jQuery provee métodos para asociar controladores de eventos a los selectores antes mencionados. Cuando sucede un evento, se ejecuta la función expresada. Al hablar de eventos se hace referencia al momento preciso en el que ocurre algo: mover el ratón sobre un elemento, seleccionar un botón, hacer clic sobre un elemento, etc.

Controladores de eventos de ratón:

- Para asignar el **evento de clic** a todos los párrafos de una página, puedes hacer esto:

```
$("p").click(function(){// definir qué acciones aplicar cuando se produzca el click sobr los párrafos });
```

- **Evento dblclick:** cuando se hace doble click, en este caso sobre el primer párrafo, se muestra un mensaje.

```
$("p:first").dblclick(function () {

        alert("Acabas de hacer doble click");

      });
```

- **Evento mouseenter:** cuando el cursor entra, en este caso, en el primer elemento <p>, se modifica la propiedad CSS “border”.

```
$("p:first").mouseenter(function () {

       $(this).css("border", "1px solid green");

 });
```

- **Evento mouseout:** cuando el cursor sale, en este caso, del primer elemento <p>, cambia la propiedad CSS “border”.

```
$("p:first").mouseout(function () {

     $(this).css("border","2px solid red");

 });
```

Controladores de eventos de teclado:

- **keypress:** captura el evento de la pulsación de una tecla.
- **keydown:** captura el evento al pulsar una de las teclas de desplazamiento (bajar).
- **keyup:** captura el evento al pulsar una de las teclas de desplazamiento (subir).

Llamadas a un evento sin que se produzca:

Hasta este momento hemos visto como se realiza una o varias acciones tras producirse un evento, pero qué tendríamos que realizar si quisiéramos que se ejecutara el código asociado a un evento sin que se este se produzca. Para esto tendremos que utilizar el método **trigger.** En este método tendremos que indicar el objeto y el evento del cual queremos que se ejecute el código asociado, de la siguiente forma:

Teniendo como ejemplo de código asociado al evento click de un botón con identificador #btn:

```
$("#btn").click(function(){<br />        /* Asignamos el valor de las variables ancho y alto los atributos width y height respectivamente de la
            $(".miniaturas").attr("width",ancho+"px");
            $(".miniaturas").attr("height",alto+"px");
       });
```

Para que se ejecute este código tendremos que escribir los siguiente:

```
$("#btn").trigger("click");
```

Debes conocer

Todos los controladores de eventos que proporciona jQuery están disponibles en la citada API.

1.2.4.- Efectos.

jQuery proporciona la posibilidad de crear efectos personalizados. A continuación, se detallarán algunos de los métodos que incluye jQuery para la creación de efectos sencillos sobre los elementos.

Mostrar y ocultar.

- **hide():** oculta los elementos coincidentes. Sintaxis: $(selector).hide(speed,callback);
- **show():** muestra los elementos coincidentes. Sintaxis: $(selector).show(speed,callback);

  En el siguiente ejemplo se ocultarán todos los elementos <p> cuando el usuario haga clic sobre el elemento con id=btn1 y se mostrarán cuando el usuario haga clic sobre el elemento con id=btn2.

```
$("#btn1").click(function(){
  $("p").hide();
});

$("#btn2").click(function(){
  $("p").show();
});
```

  Parámetros opcionales **speed** y `callback`.

  Speed especifica la velocidad de hide() / `show()`, y puede tomar los siguientes valores: "slow", "fast" o los milisegundos concretos.

  callback corresponde al nombre de una función que se ejecutará después de que ocultar o mostrar se complete.

- **toggle():** este método permite alternar entre los métodos hide() y `show()`. Sintaxis: $(selector).toggle(speed,callback);

Fading. Con los métodos de fading de jQuery se puede cambiar la propiedad **CSS** que proporciona opacidad a los elementos.

- **fadeIn():** hace que el elemento que lo recibe aparezca en la página a través del cambio de su opacidad, haciendo una transición suavizada que acaba con el valor de opacity 1. Este método solo podremos observarlo si el elemento sobre el que lo invocamos era total o parcialmente transparente, porque si era opaco al hacer un fadeIn() no se advertirá ningún cambio de opacidad. Sintaxis: $(selector).fadeIn(speed,callback);
- **fadeout():** este método hace que el elemento que lo recibe desaparezca de la página a través del cambio de su opacidad, haciendo una transición suavizada que acaba con el valor de opacity 0. Sintaxis: $(selector).fadeOut(speed,callback);
- **fadeTo():** este método permite hacer cualquier cambio de opacidad, a cualquier valor y desde cualquier otro valor. El mismo, recibe la duración deseada para el efecto, el valor de opacidad al que queremos llegar y una posible función callback. Sintaxis: $(selector).fadeTo(speed,opacity,callback);

El siguiente código nos puede servir para cambiar un conjunto de imágenes en un tiempo determinado con un efecto **fadeIn()**, **fadeOut().** Ponemos en contexto el siguiente código ejemplo. Se trata de un objeto contenedor (definido con un identificador id="contenedor" que contiene un conjunto de imágenes). Hemos definido también un vector denominado img, que almacenará el nombre de diferentes imágenes. Al activarse el siguiente código, por ejemplo al hacer click en un botón, desaparecen las imágenes en un tiempo de 500 milisegundos, se le asigna a cada imagen un origen de la matriz definida y vuelven a aparecer las imágenes en 500 milisegundos (con el origen cambiado). Si nos fijamos aplicamos el efecto mediante una función.

```
let img= ["imagen_01.jpg", "imagen_02.jpg","imagen_03.jpg"];
$("#contenedor").fadeOut(500,function(){
     $(".elementos").each(function(index) {
       $(this).attr("src","./imagenes/"+img[index]);
     });
});
$("#contenedor").fadeIn(500);
```

Desplazamiento.

- **slideDown():** se utiliza para deslizar hacia abajo un elemento. Sintaxis: $(selector).slideDown(speed,callback);
- **slideUp():** se utiliza para deslizarse hacia arriba un elemento. Sintaxis: $(selector).slideUp(speed,callback);
- **slideToggle():** alterna entre el método slideDown() y el método slideUp(), de esta forma mostrará y ocultará un elemento con efecto desplazamiento. Sintaxis: $(selector).slideToggle(speed,callback);

Encadenamiento.

Existe una técnica llamada encadenamiento, que permite ejecutar varios comandos jQuery, uno tras otro, en el mismo elemento(s). De esta forma, los navegadores no tienen que encontrar el mismo elemento más de una vez. Para realizarlo, solo es necesario añadir la acción anterior, separándolas por un punto.

Así por ejemplo, en el siguiente código el elemento con id=”p1” cambia primero a rojo, a continuación, se desliza hacia arriba, y luego se desliza hacia abajo:

```
$("#p1").css("color","red").slideUp(2000).slideDown(2000);
```

Cuando uso el encadenamiento, la línea de código puede ser bastante larga. Sin embargo, jQuery no es muy estricto acerca de la sintaxis, se le puede dar formato incluyendo saltos de línea y tabulaciones.

Recomendación

Para trabajar con el resto de efectos, se recomienda trabajar con la información disponible en la API:

1.2.5.- Modificar una página con jQuery.

En este apartado ser verá los métodos que provee jQuery para modificar nuestra página web.

Contenido - text (), html () y val():

- text () - Establece o devuelve el contenido de texto de los elementos seleccionados. Así por ejemplo, con el siguiente código, se establecerá el texto “Hello world” al elemento con id=”test1” cuando el usuario haga clic sobre el elemento con id=”btn1”.

```
$("#btn1").click(function(){
$("#test1").text("Hello world!");});
```

- html () - Establece o devuelve el contenido de los elementos seleccionados. Con el siguiente código, se establece el texto “Hello world” en negrita como valor del elemento con id=”test2” cuando el usuario haga clic sobre el elemento con id=”btn2”.

```
$("#btn2").click(function(){
$("#test2").html("<b>Hello world!</b>");});
```

- val () - Establece o devuelve el valor de los campos de un formulario. En el siguiente ejemplo, se establece el texto “Mi texto” como valor del elemento con id=”test3”, cuando el usuario haga clic sobre el elemento con id=”btn3”.

```
$("#btn3").click(function(){
$("#test3").val("Mi texto");});
```

Establecer atributos - attr():

- **attr():** Este método también se utiliza para establecer / cambiar los valores de los atributos. En el siguiente código se muestra cómo cambiar el valor del atributo href en un enlace:

```
$("button").click(function(){
$("#ww").attr("href","http://www.w3schools.com/jquery");
});
```

Añadir/Eliminar elementos HTML.

- **append ():** inserta contenido al final de los elementos HTML seleccionados.
- **prepend (): i**nserta contenido al principio de los elementos HTML seleccionados.

Añadir varios elementos nuevos con append () y prepend ()

Tanto append () como prepend () pueden tomar un número infinito de nuevos elementos como parámetros. Los nuevos elementos se pueden generar con el texto / HTML, con jQuery, o con el código JavaScript y elementos DOM.

En el código siguiente, se crean varios elementos nuevos. Los elementos se crean con texto / HTML, jQuery y JavaScript / DOM. FInalmente se añaden los nuevos elementos del texto con el método append ():

```
function AgregaTexto()
{
    var txt1="<p>Text</p>";   // Create element with HTML
    var txt2=$("<p></p>").text("Text");   // Create with jQuery
    var txt3=document.createElement("p");  // Create with DOM
    txt3.innerHTML="Text";
    $("p").append(txt1,txt2,txt3);   // Append the new elements
}
```

- **after():** inserta contenido después de los elementos HTML seleccionados.
- **before():** inserta contenido antes de la selección de los elementos HTML.
- remove(): elimina el elemento seleccionado (y sus elementos hijos). Este método acepta un parámetro, que le permite filtrar los elementos a eliminar.
- empty(): elimina los elementos hijos del elemento seleccionado.

Manipulación de CSS

- **addClass():** añade una o más clases a los elementos seleccionados.
- **removeClass():** elimina una o más clases de los elementos seleccionados.
- **toggleClass():** cambia entre la adición / eliminación de las clases de los elementos seleccionados.
- **css():** establece o devuelve el atributo de estilo.

En el siguiente ejemplo, el código siguiente devuelve el valor del tamaño de la fuente del elemento con tal id:

```
$('#h').css('fontSize');
```

Para establecer una propiedad **CSS** utiliza la siguiente sintaxis: css("propertyname","value");

En el siguiente código se establece el valor de fondo de color para todos los elementos encontrados:

```
$("p").css("background-color","yellow");
```

Establecer múltiples propiedades CSS: para establecer múltiples propiedades **CSS**, utiliza la siguiente sintaxis:

```
css({"propertyname":"value","propertyname":"value",...});
```

Definir eventos de forma dinámica:

Ahora nos situaremos en el caso de que insertemos elementos de forma dinámica en nuestra página y queramos asignar un comportamiento determinado a un evento, para ello tendremos que indicar el elemento y el evento. A continuación, dejamos un ejemplo donde se define el comportamiento del elemento p al hacer doble click en los elementos contenidos dentro de los elementos con clase información.

```
$(".informacion").on("dblclick", "p", function() {
      $(this).css("border","red 10px solid");
});
```

1.3.- Propiedades de los elementos.

El modelo DOM que vimos en el apartado anterior establece una relación jerárquica entre los elementos (objetos) que forman parte de un documento HTML y la relación de éstos con el navegador.

Cada uno de estos elementos tienen una serie de propiedades que se podrán modificar en función de las acciones del usuario pero para hacerlo es necesario que cada elemento esté identificado de forma unívoca.

Reflexiona

¿Recuerdas cómo se podía identificar un elemento del HTML de forma unívoca?

Mostrar retroalimentación

Había que nombrar el elemento mediante el atributo id. Por ejemplo: <p id="primer_parrafo">.

Cuando un elemento está identificado mediante un nombre único en todo el documento, el lenguaje utilizado para implementar la interactividad puede localizar el elemento fácilmente y acceder a sus propiedades, o atributos, y cambiarlas. Esto se debe a que, en el modelo DOM, los atributos de una etiqueta HTML son traducidos por el navegador en propiedades de un objeto.

Debes saber

En el siguiente enlace puedes ver todos los objetos (elementos) que forman parte de un documento con sus propiedades y con los métodos o acciones que se pueden realizar sobre el mismo.

Modelo de objetos de documento (HTML), Nivel 1.

Para saber más

Desde el siguiente enlace puedes acceder a las especificaciones del W3C de los tres niveles del modelo DOM.

Modelo de objetos de documento (HTML), Nivel 1, 2 y 3.

Pero en un documento HTML puede haber un número muy elevado de elementos div o de elementos input, o cualquier otro elemento repetido cada uno con su identificador exclusivo. En ese caso ¿cómo identifica este modelo a cada uno de esos elementos? La respuesta es muy simple, utiliza un vector. En en la siguiente presentación puedes ver un ejemplo comentado de las formas que tenemos de acceder a un elemento del modelo de objetos de documento.

Resumen textual alternativo

Recomendación

Los siguientes son enlaces a las páginas de sus autores que explican, con ejemplos, el modelo de objetos de documento.

Introducción al Modelo de Objetos de Documento (DOM).

¿Qué es el DOM?

Anexo.- Licencias de recursos.

Licencias de recursos utilizados en la Unidad de Trabajo.

Recurso (1)	Datos del recurso (1)	Recurso (2)	Datos del recurso (2)
	Autoría: Stockbyte. Licencia: Uso educativo no comercial para plataformas públicas de Formación Profesional a distancia. Procedencia: CD-DVD Num. CD165.		Autoría: Stockbyte. Licencia: Uso educativo no comercial para plataformas públicas de Formación Profesional a distancia. Procedencia: CD-DVD Num. V43.
	Autoría: JohnManuel. Licencia: Creative Commons Attribution-Share Alike 3.0 Unported. Procedencia: Montaje sobre: http://upload.wikimedia.org/wikipedia/commons/thumb/9/90/DocumentObjectModel.svg/1000px-DocumentObjectModel.svg.png		

Condiciones y términos de uso de los materiales

Historial de actualizaciones

Versión: 01.02.01	Fecha de actualización: 10/04/23	
Actualización de materiales y correcciones menores.		
Versión: 01.02.00	**Fecha de actualización: 23/03/21**	**Autoría: Jesús Moreno Ortiz**
Ubicación: 1.3 **Mejora (tipo 2):** Se cambia presentación por video. **Ubicación:** 1.2.4 y 1.2.5 **Mejora (tipo 2):** Se añaden códigos ejemplos **Ubicación:** 1.2.3. **Mejora (tipo 2):** Se añade contenido relacionado con el método trigger y ejemplo. **Ubicación:** 1.2.2. **Mejora (tipo 2):** Se añaden nuevos ores y métodos de selección y se incluye código ejemplo. **Ubicación:** 1.2.1 **Mejora (tipo 1):** Se unifica el estilo del código **Ubicación:** 1.2.1. **Mejora (tipo 1):** Pequeñas erratas.		
Versión: 01.01.00	**Fecha de actualización: 08/05/14**	**Autoría: Eliana Yemina Manzano Fernández**
Actualización con jQuery.		
Versión: 01.00.00	**Fecha de actualización: 08/05/14**	
Versión inicial de los materiales.		