

FrontEnd ADA

JQuery

Introducción a la librería JS

Maria Belen Alegre

JQuery

Como hemos comentado anteriormente una de las librerías de JavaScript más utilizada resulta ser JQuery, veremos con más detalle su integración en nuestras aplicaciones y sus principales características. jQuery fue creada inicialmente por John Resig.

Con JQuery podemos principalmente:

- Acceder a elementos en un documento.
- Modificar la apariencia de una web.
- Alterar el contenido de un documento.
- Responder a una interacción del usuario.
- Animar cambios en un documento.
- Recoger información del servidor sin refrescar la página.
- Tareas diversas de Javascript.

Utilizar jQuery

Existen distintas formas de incluir la librería jQuery en nuestra página web, revisaremos las variantes a continuación.

Opción 1:

1. En primer lugar se debe bajar la versión más actual de jQuery. Esto lo podemos hacer desde la propia web de jQuery haciendo click sobre el botón Download (jQuery).
2. Una vez descargado el archivo la mejor práctica es guardarlo en una carpeta que contenga todo el código JavaScript del proyecto (por ejemplo js). Luego en la cabecera del código de la web se añade jQuery como lo haríamos con cualquier archivo de Javascript.

Ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">

<head>
```

```

<title>TITULO DE LA WEB</title>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<link rel="stylesheet" type="text/css" href="css/default.css" />

<script type="text/javascript" src="js/jquery-1.3.2.min.js "></script>

</head>

<body>

// Contenido de la Web

</body></html>

```

Incluimos la librería
descargada

Opción 2:

Como opción a la descarga de la librería, es posible incluirla directamente desde Google.

Ejemplo

```

</html><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">

<head>

<title>TITULO DE LA WEB</title>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<link rel="stylesheet" type="text/css" href="css/default.css" />

<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.3/jquery.min.js"></script>

</head><body>

// Contenido de la Web

</body></html>

```

Incluimos la librería directamente de Google

Opción 3:

Utilizar siempre la última versión de jQuery, para esto simplemente nos conectamos con la página oficial de la siguiente manera:

```
<script type="text/javascript" src="http://code.jquery.com/jquery-latest.js"></script>
```

Incluimos la última versión directamente del sitio oficial

Nota: versiones de jQuery y compatibilidades, es importante tener en consideración a la hora de elegir la versión de jQuery a utilizar la compatibilidad, la última versión 2.x, posee la misma API que la 1.x, sin embargo, no soporta Explorer 6,7,8. Por lo tanto si el upgrade es para un sitio que necesariamente debe estar disponible para estas versiones del navegador es recomendable no realizar el upgrade.

Agregar código jQuery

Existen dos instancias en las cuales se puede desear añadir código jQuery, al momento de cargar el sitio web, o simplemente dentro de ella una vez cargada, en el ejemplo siguiente vamos a probar como cargar inicialmente el efecto creado con jQuery.

Ejemplo

Añadiremos la clase "fondo-destacado" a un div con id ="contenido".

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">
<head> <title>GenerandoIT</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" type="text/css" href="css/default.css" />
<script type="text/javascript" src="js/jquery-1.3.2.min.js "></script>
</head>
<body>
<h3>Esta es una prueba de jQuery</h3>
<div id="contenido">
```

```

<p>Testeando jQuery </p>

</div>

</body></html>

```

Sabiendo que dentro de la hoja de estilos default.css, contiene el siguiente elemento de estilo:

```
.fondo-destacado { background-color: red; }
```

Para añadir un código de jQuery que se ejecute en cuanto se cargue el documento de nuestra web (y por lo tanto está ya disponible el div con el id “contenido” de nuestro ejemplo) añadimos las siguientes líneas:

```

<script type="text/javascript">

    $(document).ready(function() {
        // líneas de código de jQuery
    });
</script>

```

Cuando el documento (document) esté listo (ready) ejecutar la función que sigue a continuación.

Este comienzo es típico de JQuery y lo que dice es que cuando el documento (document) esté listo (ready) se ejecute la función que sigue a continuación. El documento es el DOM (Document Object Model). Merece la pena distinguir esto de la manera tradicional utilizada en Javascript:

```

<script type="text/javascript">

    window.onload = function() {

```

Con window.onload lo que hacíamos es que evitar que ejecute la función hasta que no se cargara la web completa (no sólo el DOM), es decir, todas las imágenes y recursos externos. Esto implicaba que si había alguna imagen de carga muy lenta cualquier evento gestionado con Javascript no iba a funcionar hasta que no se cargara esa imagen. Este impedimento lo evitamos con el \$(document).ready de JQuery.

Ahora veamos el código completo del ejemplo:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">

```

```
<head>
```

```
<title>TITULO DE LA WEB</title>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
```

```
<link rel="stylesheet" type="text/css" href="css/default.css" />
```

```
<script type="text/javascript" src="js/jquery-1.3.2.min.js "></script>
```

```
<script type="text/javascript">
```

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

Se ejecuta la función cuando la página está lista.

```
$('#contenido').addClass('fondo-destacado');
```

```
});
```

```
</script>
```

Método. addClass() añadimos una clase de CSS a la parte de la página que hemos seleccionado anteriormente, es decir al Div con id="contenido". Su único parámetro es el nombre de la clase, escrito entre comillas:
 .addClass('fondo-destacado');

```
</head><body>
```

Selector. Esto se hace con el constructor \$().

```
<h3>Esta es una prueba de jQuery</h3>
```

```
<div id="contenido">
```

```
<p>Esta es una prueba de jQuery en la que vamos a añadir la clase "fondo-destacado" al Div con el id="contenido"</p>
```

```
</div>
```

```
</body></html>
```


Elementos jQuery

Selectores

Una forma de permitirnos elegir un elemento (o varios) entre todos los que tenemos en nuestro documento HTML. ¿Para qué? Para luego poder aplicar sobre los elementos seleccionados diversas funciones.

Es decir, jQuery utiliza el poder de los selectores para acceder de una manera rápida y sencilla a un elemento o grupo de elementos del DOM (Document Object Model) y luego poder aplicar sobre los mismos cualquier tipo de instrucción, evento, animación, etc. Por ejemplo, para aplicar la clase “enlace” a todos los elementos “a” que se encuentren dentro de un elemento “p”.

Ejemplo: `$('p a').addClass('enlace');`



Selecciona todos los a descendientes de un p.

`$('p')` ← Selecciona todos los párrafos del documento.

`$('#nombre-id')` ← Selecciona el elemento con el id = 'nombre-id'.

`$('.nombre-clase')` ← Selecciona todos los elementos de esa clase.

`$('p,a')` ← Selecciona todos los elementos a y p del documento.

`$('li.nombre_clase')` ← Selecciona todos los de esa clase.

Selectores propios de jQuery

A la amplia variedad de selectores propios de CSS jQuery añade sus propios selectores. Como característica los distingue que siempre comienzan por dos puntos (:)

Selectores Posicionales

Estos selectores están basados en las relaciones posicionales entre elementos (como veíamos antes en ejemplo de la estructura del DOM).

`$('p:first')` ← Selecciona el primer elemento <p> del documento.

```
$('#img[src$=.png]:first')
```

Selecciona la primera imagen del documento del tipo png.

Selectores de Formularios

Cuando trabajemos con formularios jQuery nos ofrece una serie de selectores propios que nos permiten seleccionar de manera sencilla el elemento preciso.

```
$(':text')
```

```
$(':checkbox')
```

```
$(':radio')
```

```
$(':image')
```

```
$(':submit')
```

Selecciona todos los elementos <input> con un tipo de atributo del nombre del selector.

```
$(':reset')
```

```
$(':password')
```

```
$(':file')
```

Selecciona todos los elementos <input> con un tipo de atributo del nombre del selector.

Combinación de selectores

```
$('radio:checked'),$(':text:disabled')
```

Filtros

Son aquellos que concretan una selección de elementos ya realizada aplicando una selección adicional que limita la selección anterior (por ejemplo, hemos seleccionado previamente todos los elementos div y ahora queremos sólo el cuarto div de esa selección). Los filtros se distinguen así de los selectores que encuentran otros elementos que guardan relación con los elementos ya seleccionados.

.not(), lo utilizaremos siempre que deseemos seleccionar aquellos elementos que no cumplen con un determinado filtro situado dentro de los paréntesis del .not().

Ejemplo

```
$("#div").not(".verde, #azul").css("border-color", "red");
```

Primero selecciona todos los div del documento y luego selecciona aquellos que no tengan la clase 'verde' ni el div 'azul' y les aplica el estilo correspondiente de CSS.



.filter()

Otra manera de filtrar una selección de elementos es reducirla a aquellos que sí cumplen con un filtro adicional, y lo hacemos a través de .filter()

Ejemplo:

Deseamos cambiar el color de fondo de todos los divs y después poner un borde alrededor de sólo los que tienen la clase 'centro'.

```
$("#div").css("background", "#c8ebcc")
```

```
.filter(".centro")
```

```
.css("border-color", "red");
```

Resultado:



.slice(comienzo, final)

Hay ocasiones en las que queremos obtener una porción de una determinada selección de elementos basada en la posición de estos elementos dentro de la selección realizada. Entonces utilizamos .slice(), que creará una nueva selección. Admite dos parámetros, un número que indica el inicio (empezando por cero) donde vamos a realizar el corte y otro que indica el final no incluido. Este último valor es opcional.

Ejemplo:

```
$('p').slice(2,3);
```

El primer número (donde se empieza a cortar la selección inicial) es un dos porque se empieza a contar desde cero: 0, 1, 2 es decir, empezamos a contar por el tercer elemento. Para el final no incluido pone 3, es decir el cuarto elemento (0,1,2,3) por lo que de todos los p del documento solo se selecciona el tercero.

.eq() Este filtro reduce el grupo de elementos seleccionados a un elemento único. Su argumento (lo que va entre los paréntesis) es la posición del elemento dentro del grupo de elementos previamente seleccionados (que puede ser un número entre 0 y la longitud completa del número de elementos menos uno, al haber empezado en cero).

Otro tipo de filtros muy utilizados suelen ser los de visibilidad.

.hidden() Este filtro selecciona todos los elementos que están definidos como ocultos. Los elementos pueden considerarse ocultos por distintas condiciones que detallamos a continuación:

- Tienen un valor CSS *display:none*.
- Son elementos de formulario con `type = hidden`.
- Su anchura y altura son explícitamente 0.
- Un elemento padre está oculto, por lo que el elemento no se muestra en la página.

A continuación, revisaremos un ejemplo del uso de este filtro.

Ejemplo:

Considerando en una hoja de estilos el siguiente código:

```
div { width:70px; height:40px; background:#ee77ff; margin:5px; float:left; }
```

```
span { display:block; clear:left; color:red; }
```

```
.starthidden { display:none; }
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html><head>
```

```
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
```

```
</head><body>
```

```
<span></span>

<div></div>

<div style="display:none;">Hider!</div>

<div></div>

<div class="starthidden">Mostrar</div>

<div></div>

<form>

  <input type="hidden" />

  <input type="hidden" />

  <input type="hidden" />

</form>

<span></span>

<script>

var hiddenEls = $("body").find(":hidden").not("script");

$("span:first").text("Encuentra " + hiddenEls.length + " elementos en total.");

$("div:hidden").show(3000);

$("span:last").text("Found " + $("input:hidden").length + " hidden inputs.");

</script>

</body></html>
```

El script se ubicó al final, para ser ejecutado luego de cargar completamente la página.

Ahora analicemos cada elemento.

<script>

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

← Agregado para esperar a que se cargue la página antes de ejecutar.

`var hiddenEls = $("body").find(":hidden").not("script");`

Variable local, contendrá la cantidad de elementos ocultos.

Selector. Elementos del body.

Encontrar elementos, considerando el filtro .hidden.

Excluir elementos cuyo atributo sea script. **Nota:** esta parte es sólo necesaria si se carga el script dentro del cuerpo.

```
$("#span:first").text("Found " + hiddenEls.length + " hidden elements total.");
```

En el primer span mostrar el texto

```
$("#div:hidden").show(3000);
```

Mostrar los elementos ocultos utilizando el efecto. El parámetro es la duración del efecto.

```
$("#span:last").text("Found " + $("#input:hidden").length + " hidden inputs.");
```

```
});
```

```
</script>
```

Nota: para incluir el código jQuery en la parte superior de la página se debe utilizar el código anterior. Al adicionarse el método `ready()`, nos aseguramos que se ejecute luego de cargar completamente la página.

.visible() De forma complementaria este filtro selecciona todos los elementos visibles del documento, son considerados visibles todos aquellos elementos que ocupan espacio, tienen un ancho y alto mayor a cero.

Selectores por jerarquía

.find()

Esta función retorna los descendientes de cada elemento en el conjunto actual de elementos coincidentes, filtrados por un selector, objeto jQuery o elemento.

```
$( "p" ).find( "span" ).css( "color", "red" );
```

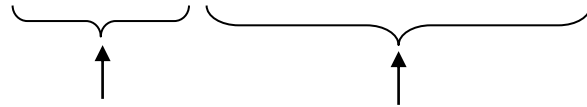
Busca los elementos **span** dentro del **p**

Aplica el cambio de color en la fuente.

.children()

Esta función retorna los elementos secundarios de cada elemento en el conjunto de elementos coincidentes, opcionalmente filtrados por un selector.

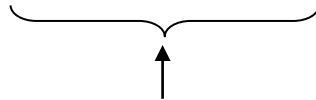
```
$( "div" ).children().css( "border-bottom", "3px double red" );
```



Obtengo los hijos del div

Aplico propiedad css

```
$( "div" ).children( ".selected" ).css( "color", "blue" );
```



Obtengo sólo los hijos del div que tienen la clase selected

Nota: El método **.children ()** difiere de **.find ()** en que **.children ()** solo recorre un único nivel en el árbol DOM mientras que **.find ()** puede recorrer múltiples niveles para seleccionar elementos descendientes (nietos, etc.) también.