- Es la librería (*framework*) JavaScript para escribir menos y hacer más…
- Es una forma de convertir el desarrollo cliente de una aplicación web en algo mucho más rápido y sencillo, facilitando la interacción con los elementos del árbol de documento (DOM), el manejo de eventos, el uso de animaciones, etc.

Para poder utilizar esta librería lo primero que tendremos que hacer será incluir su código en el <head> de la página web.

Podemos descargar el script desde su página web, subirlo a nuestro servidor, y ejecutarlo con la etiqueta script:

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

También cargarla directamente desde los CDN (repositorios) que mantienen **Google, Microsoft** o **jQuery**:

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.11.0.min.js"></script>
<script src="http://code.jquery.com/jquery-1.11.0.min.js"></script>
```

Para actualizar a futuras versiones: http://jquery.com/download/

*Nota: si vas a utilizar librerías jQuery 1.9 y posteriores, es muy recomendable que uses también el plugin **jquery-migrate**, que asegura que funcionen órdenes antiguas jquery usando las nuevas librerías:*

```
<script src="http://code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
```

Como jQuery trabaja sobre los objetos del DOM, lo primero que debemos hacer es asegurarnos que el árbol DOM ha sido construido completamente (para evitar que cualquier código jQuery se ejecute antes de que el documento se termine de cargar):

```
$(document).ready(function() {
   // sentencias jQuery
 });
```

O también y mucho más corto:

```
$(function() {
   // sentencias jQuery
 });
```

Sintaxis jQuery

La sintaxis de jQuery está hecha a medida para la selección de los elementos HTML y realizar alguna acción sobre el elemento.

La función jQuery **$()** sirve para hacer varias cosas, según los parámetros que le pasemos. El uso más simple es seleccionar elementos o grupos de elementos de una página web.

Veamos la sintaxis básica: `$(selector).accion();`

- el signo de dólar para definir jQuery
- un "selector" para localizar elementos HTML
- una función o método, *accion()*, para realizar operaciones sobre el elemento HTML

Nota: $() es una forma corta de invocar la función jQuery. También podríamos haber escrito la línea de esta manera: `jQuery(selector).accion();`

Selectores

Lo primero que vamos a hacer cuando trabajamos con jQuery es seleccionar elementos sobre los que trabajar. Para ello se utiliza la función jQuery, pasando como argumento una cadena con un selector, la mayoría de los cuales utilizan una sintaxis similar a la de CSS 3 (*nos permite usar selectores CSS 3 en navegadores antiguos como IE 6, 7 y 8*).

Selectores básicos

- Selector de tipo o etiqueta: selecciona todos los elementos con el tipo de etiqueta

```
$("div")
```

- Selector de clase: selecciona todos los elementos con la clase indicada

```
$("div.entrada")
```

- Selector de identificador: selecciona el elemento con el identificador

```
$("div#cabecera")
```

- Grupos de selectores: se puede combinar el resultado de varios selectores distintos separándolos con comas

```
$("p,div,a")
```

Selectores jerárquicos.

Selecciona todos los elementos descendientes de otro elemento

```
$('div img')
```

```
$('div > img')
```

```
$('div + img')
```

Selectores de atributos

- Selector de atributo: selecciona elementos que tengan un cierto atributo

```
$("a[rel]")
```

$("a[rel='nofollow']")

- Aquellos cuyo valor empieza, termina o contiene cierta cadena

$("a[href^='http']")

$("a[href$='.com']")

$("a[href*='geek']")

Selectores por posición

$('div:first') // Selecciona el primer DIV.

$('div:last') // Selecciona el último DIV.

$('div:even') // Selecciona los DIV impares (comienza a contar desde 1, 3, 5,..)

$('div:odd') // Selecciona los DIV pares (comienza a contar desde 0, 2, 4,..)

$('div:eq(4)') // Selecciona el quinto DIV, (comienza a contar desde 0)

Selectores de filtrado personalizado

$("div:contains('I love HTML5')") // Selecciona todos los DIV que contengan el texto indicado, directamente o en uno de los hijos

$("div:has(h2)") // Selecciona todos los DIV que contengan al menos un H2

$('div:not(.entrada)') // Selecciona los DIV que no tengan un class=”entrada”.

this

$(this) // al seleccionar usando this lo que se hace es seleccionar al elemento sobre el que actuamos.

Eventos en jQuery

Los eventos son la base para crear la interacción con el usuario, algo fundamental en las páginas que usan jQuery.

El trabajo básico con eventos es siempre el mismo:

1. Seleccionamos el elemento
2. Asociamos el evento al elemento
3. Escribimos la función que se ejecutará cuando se lance en evento

```
$(function(){
   $("a").click(function(){
     alert("Este mensaje se muestra cuando el usuario pincha en el enlace");
     });
 });
```

Como vemos, el tipo de evento se define a partir de una **función click()** o similares. Existen diferentes tipos de funciones que implementan cada uno de los eventos normales, como dblclick(), focus(), keydown(), etc.

1) Eventos relacionados con el ratón

click() //Sirve para generar un evento cuando se produce un clic en un elemento de la página.

dblclick() //Para generar un evento cuando se produce un doble clic sobre un elemento.

mousedown() //Para generar un evento cuando el usuario hace clic, en el momento que presiona el botón e independientemente de si lo suelta o no. Sirve tanto para el botón derecho como el izquierdo del ratón.

mouseup() //Para generar un evento cuando el usuario ha hecho clic y luego suelta un botón del ratón. El evento mouseup se produce sólo en el momento de soltar el botón.

mouseenter() //Este evento se produce al situar el ratón encima de un elemento de la página.

mouseleave() //Este se desata cuando el ratón sale de encima de un elemento de la página.

mouseover() //Se produce cuando el ratón está sobre un elemento, pero tiene como particularidad que al pasar el ratón sobre un elemento secundario (hijo) se repite el evento al entrar y al salir de éste.

mouseout() //Se desata cuando el usuario sale con el ratón de la superficie de un elemento (igual particularidad de repetir al pasar por hijo que mouseover).

mousemove() //Evento que se produce al mover el ratón sobre un elemento de la página.

hover() //Esta función en realidad sirve para manejar dos eventos, cuando el ratón entra y sale de encima de un elemento (permite pasar 2 funciones, la primera será ejecutada *mouseenter*, y la segunda *mouseleave*).

resize() //cuando cambiamos el tamaño de la ventana.

scroll() //Cuando se hace scroll; se aplica a los objetos ventana, frames y elementos con la propiedad overflow CSS puesta en estado “scroll”.

toggle() //genera comportamientos de cambio de estado al pinchar sobre un elemento (ej. mostrar/ocultar).

```
$('#target').toggle(function() {
        alert('Primer click');
        }, function() {
        alert('Segundo click');
        }, function() {
        alert('tercer click');
});
```

Nota: toggle desaprobado en jQuery 1.8, y se elimina en la versión 1.9. (se puede seguir usando con jquery mígrate)

2) Eventos relacionados con el teclado

keydown() //momento que se presiona una tecla del teclado, independientemente de si se libera la presión o se mantiene.

keypress() //Este evento ocurre cuando se digita un carácter. Se ejecuta una vez, como respuesta a una pulsación e inmediata liberación de la tecla, o varias veces si se pulsa una tecla y se mantiene pulsada.

keyup() //se ejecuta en el momento de liberar una tecla .

3) Eventos combinados teclado o ratón

focus() //el elemento gana el foco de la aplicación

blur() //el elemento que acaba de perder el foco

Método bind()

Este método es uno de los principales de jQuery desde su inicio.

```
$("a").bind("click", function(){ …. });
```

Usando bind podemos asociar varios eventos simultáneamente.

```
$("a").bind("mouseenter mouseleave dblclick", function() { …. });
```

Además, podemos generar diferentes acciones en cada evento:

```
$("#boton").bind({ click: function(){ …. }, mouseenter: function(){ …. } });
```

*[**live()** sirve para lo mismo que bind(), pero además afectará a todos los elementos que casen con el selector en el futuro: porque los crees dinámicamente con jQuery o asignes dinámicamente una clase CSS, o que traigas por Ajax un contenido. Actualmente obsoleto.]*

Método on()

El objetivo de on() es reemplazar a los métodos bind y live desde la versión jQuery 1.7.

La sintáxis es igual que bind:

```
$("#boton").on({ click: function(){ …. }, mouseenter: function(){ …. } });
```

Método one()

Semejante a bind, pero el evento sólo se ejecuta una vez.

```
$('#boton').one('click', function() {
  alert('Este mensaje se muestra sólo una vez');
});
```

jQuery y CSS

El método **css()** sirve tanto para recibir el valor de un atributo CSS como para asignarle un nuevo valor y su funcionamiento depende de los parámetros que podamos enviarle.

$(selector).css(“nombre_propiedad_css”, “valor”)

Podemos cambiar el valor de una propiedad CSS mediante jQuery:

$("#micapa").css("background-color", "green");

Cambiar varios atributos CSS al mismo tiempo

$(selector).css(**{**“nombre_propiedad_css”:“valor”, “nombre_propiedad2_css”:“valor2”**}**)

Ejemplo: $("#micapa").css({"background-color":"yellow","font-size":"200%"});

Obtener y asignar dimensiones

Modificar el ancho y el alto es tan común, que jQuery nos permite cambiarlos directamente con los métodos width() y height(): $("#micapa").width(800);

Añadir y quitar clases CSS

Para añadir una clase usamos el método **addClass(“nombre_clase”)**, donde tenemos que pasarle una cadena con el nombre de la clase CSS que queremos añadir. De igual forma para quitar la clase usamos el método **removeClass(“nombre_clase”)**.

Además, **toggleClass(“nombre_clase”)** permite conmutar clases, de forma, que si el elemento tiene ya la clase, la elimina y viceversa.

```
$("div").addClass("miclase");

$("div").removeClass("miclase");

$("div").toggleClass("miclase");
```

Nota: tener en cuenta que estamos manipulando CSS, por lo tanto ¡ojo! con la especificidad (etiquetas, clases, id). Una clase sola (.miclase) no puede cambiar una propiedad que ya tiene un id (#midiv); pero si puede una combinación: #midiv.miclase

Manipulación de elementos HTML

jQuery nos permite modificar el contenido de los elementos HTML sin recargar la página.

Sustituir contenido

html(): reemplaza el contenido HTML de un elemento.

```
$("#btnAddHtml").click(function() {
    $("#parrafoConHTML").html("codigo <b>HTML</b>");
});
```

text(): Es similiar a html() pero en este caso está obviando las etiquetas HTML.

```
$("#btnAddText").click(function() {
    $("#parrafoConHTML").text("codigo <b>HTML</b>");
})
```

Mover y añadir elementos

append(): Se utiliza para añadir código HTML dentro del elemento seleccionado, pero al final.

```
$("#btnAppend").click(function(){
    $("#spanParaAppend").append("<p>Código añadido a través de <b>append()</b></p>");
})
```

appendTo(): Elegimos un elemento (#div1) y se lo añadimos dentro a otro elemento distinto (#otrodiv) al final de su contenido.

```
$("#btnAddAppend").click(function(){
    $("#div1").appendTo("#otrodiv");
})
```

prepend(): "antepone" contenido en el interior del elemento

```
$("#btnPrepend").click(function(){
      $("#spanParaPrepend").prepend("<p>Código añadido a través de <b>prepend().</b></p>");
})
```

prependTo(): Elegimos un elemento (#div1) y se lo añadimos dentro a otro elemento distinto (#otrodiv) al principio de su contenido.

```
$("#btnAddPrepend").click(function(e){
     $("#div1").prependTo("#otrodiv");
})
```

after(): Permite insertar código HTML a continuación del elemento seleccionado.

```
$("#btnAfter").click(function(){
    $("#spanAfter").after("<p> Insertado con <u>after</u></p>");
})
```

insertAfter(): inserta elementos (#div1) después de aquel (#otrodiv) tomado como referencia.

```
$("#btnInsertAfter").click(function(){
    $("#div1").insertAfter("#otrodiv");
})
```

before(): Coloca código HTML antes del elemento seleccionado.

```
$("#btnBefore").click(function(){
    $("#spanBefore").before("<p> Insertado con <u>before()</u>.</p>");
})
```

insertBefore(): inserta elementos (#div1) antes de aquel (#otrodiv) tomado como referencia.

```
$("#btnInsertBefore").click(function(){
    $("#div1").insertBefore("#otrodiv");
})
```

Envolver y desenvolver elementos

wrap(): sirve para "envolver", con un elemento padre, uno o varios elementos HTML.

```
$("#btnWrap").click(function(){
    $("#parrafoA").wrap("<a href='#'></a>");
});
```

wrapAll(): En este caso envuelve todos los elementos en un mismo elemento padre.

```
$("#btnWrapAll").click(function(){
    $("li").wrapAll("<div style='border:solid 1px black';></div>");
});
```

Eliminar elementos

remove(): Elimina todo el elemento(s) seleccionado.

```
$("#btnEmpty").click(function(){
      $("#parrafoA").remove();
});
```

empty(): Vacía el contenido de los elementos.

```
$("#btnEmpty").click(function(){
        $("#parrafoA").empty();
});
```

Duplicar elementos

clone(): se utiliza para duplicar elementos.

```
$("#btnClone").click(function(){
      $(this).clone().insertAfter(this);
});
```

Reemplazar elementos

replaceWith(): reemplaza los tags HTML de los elementos por los indicados en la llamada.

```
$("#btnReplaceWith").click(function(){
   $("p").replaceWith("<b> Nuevo contenido </b>");
});
```

Manipulación de atributos HTML

Invocando el método **attr()** podemos acceder a los atributos de una etiqueta HTML y modificar sus valores:

```
$(selector).attr("nombre_atributo", "nuevo_valor");
```

Además, podemos modificar varios atributos de una sola vez o si esos atributos no existían, simplemente los crea con los valores enviados:

```
$('a').attr({'title': 'Titulo nuevo', 'href': 'http://www.mipagina.com', 'style': 'color: #f80'});
```

También podemos cambiar toda la hoja de estilos de una página web con un solo clic:

```
<link rel="stylesheet" type="text/css" href="estilos/warm.css" id="miscss">
```

```
$("#miscss").attr("href", "estilos/cool.css");
```

Traversing

Sirve para recorrer el árbol del DOM y manipular los elementos que queramos: elementos padres, hijos o hermanos respecto a un elemento dado.

Seleccionar padres

parent(): selecciona solo el padre más cercano.

```
$(".b1").parent().css("border","1px solid red");
```

parents(): selecciona todos los padres del elemento (incluído body y html)

```
$(".c").parents().css("border","1px solid black");
```

Seleccionar hijos

children(): selecciona solo los hijos directos.

```
$(".a").children().css("border","1px solid green");
```

find(*parametro*): selecciona todos los hijos indicados.

```
$(".b3").find("div").css("border","1px solid blue");
```

Seleccionar hermanos

next() y **prev():** selecciona solo un hermano siguiente / anterior.

```
$(".b2").next().css("border","1px solid grey");
```

nextAll() y **prevAll():** selecciona todos los hermanos siguientes / anteriores.

```
$(".b3").nextAll().css("border","1px solid teal");
```

siblings(): selecciona todos los hermanos anteriores y siguientes.

```
$(".b4").siblings().css("border","1px solid red");
```

Efectos

jQuery nos provee de un serie de efectos visuales.

show(): Modifica los atributos alto, ancho y transparencia, partiendo de 0.

$(selector).show("*velocidad*");

velocidad: Determina el tiempo en el que se realizará el efecto. Puede ser "slow" (lento), "normal", "fast" (rápido); o se le puede pasar un valor numérico en milisegundos.

hide(): Modifica los atributos alto, ancho y transparencia, partiendo de los valores actuales hasta llegar a 0.

$(selector).hide("*velocidad*");

toggle(): permite cada vez que se ejecute cambiar de estado la visibilidad del elemento HTML, es decir si está visible pasa a oculto y si se encuentra oculto pasa a visible.

$(selector).toggle("*velocidad*");

slideDown(): Modifica los atributos alto y transparencia, partiendo de 0. Es similar a "show", salvo que no modifica el ancho, creando un efecto de "cortinilla".

```
$(selector).slideDown("velocidad");
```

slideUp(): Modifica los atributos alto y transparencia, partiendo de los actuales, hasta llegar a 0. Es similar a "show", salvo que no modifica el ancho, creando un efecto de "cortinilla".

```
$(selector).slideUp("velocidad");
```

slideToggle(): cada vez que se ejecute cambia los atributos alto y transparencia de 0 a su valor actual y viceversa.

```
$(selector).slideToggle("velocidad");
```

fadeIn(): Modifica el atributo transparencia desde 0 hasta su valor actual.

```
$(selector).fadeIn("velocidad");
```

fadeOut(): Modifica el atributo transparencia desde el valor actual, hasta llegar a 0.

```
$(selector).fadeOut("velocidad");
```

fadeToggle(): cada vez que se ejecute la transparencia de 0 a su valor actual y viceversa.

$(selector).fadeToggle("velocidad");

fadeTo(): Modifica el atributo transparencia a un valor especifico.

$(selector).fadeTo("velocidad", *transparencia*);

transparencia: Un numero de 0 a 1 que indica la visibilidad del elemento.

Efectos personalizados con animate()

Para realizar animaciones en propiedades CSS.
Sólo permite propiedades con valores numéricos.

$(selector).animate({"*propiedades*"},"*velocidad* (opcional)","*easing*" (opcional)");

propiedades: Una o más propiedades con valor numérico en CSS.
velocidad: "slow", "normal", "fast", o en milisegundos.
función de animación (easing): cómo se realizará la animación, suave al principio y rápida al final ("swing", por defecto), o igual todo el tiempo ("linear").

$(selector).animate({"width":"760px","height":"290px"},"slow");

Los estilos se definen con las referencias JavaScript (“fontSize”), o por sintáxis CSS (“font-size”).

Posibles valores de estilo CSS son: margin, padding, height, width, maxHeight, maxWidth, font, fontSize, bottom, left, right, top.

[ver referencia completa: http://www.w3schools.com/jquery/eff_animate.asp]

Color animation jQuery

Con *animate()* no podemos hacer animaciones de color (no tienen valores numéricos), pero existe un plugin - *Color animation* - que nos permitirá hacer animaciones de color directamente con el método animate(), basta con incluirlo en nuestras páginas:

```
<script src="jquery.animate-colors-min.js"></script>
```

(hay 2 versiones: jQuery 1.7 e inferior; y jQuery 1.8 y superior)

Propiedades soportadas: *color, backgroundColor, borderColor, borderBottomColor borderLeftColor, borderRightColor, borderTopColor, outlineColor* (tb. en formato CSS).

```
$(selector).animate({"background-color":"#400101","color":"#99FF00"},"slow");
```

[Nota: incluído en librería jquery UI]

Cola de efectos jQuery

Cuando invocamos varios efectos, éstos se van introduciendo en una cola de efectos predeterminada, llamada "fx" y se van ejecutando automáticamente, uno detrás de otro, con el orden en el que fueron invocados.

```
var capa = $("#micapa");

capa.fadeOut();

capa.fadeIn();

capa.slideUp();

capa.slideDown();
```

Las funciones de efectos no se ejecutan todos a la vez, sino que se espera que acabe el anterior antes de comenzar la siguiente. Por suerte, jQuery hace todo por su cuenta para gestionar esta cola.

Intervalos de espera en la cola

delay() sirve para generar un intervalo de espera entre la ejecución de funciones de la cola de efectos.

```
capa.fadeOut();
capa.delay(1500);
capa.fadeIn();
capa.delay(1500);
```

Detener efectos

stop es el método que usaremos para detener la animación actual.

stop(): detiene el efecto actual, pero continúa con los siguientes efectos encolados.

```
$("#elemento").stop();
```

stop(true): detiene el efecto actual y los siguientes efectos encolados (se detiene todo).

stop(false, true): termina el efecto actual y salta para mostrar el final de ese efecto. Luego continúa con los siguientes efectos encolados.

Recursos

http://jquery.com/

http://librojquery.com/

http://mundogeek.net/archivos/2010/04/21/tutorial-rapido-de-jquery/

http://www.tutorialjquery.com/

http://www.javascriptya.com.ar/jquery/

http://www.axtro.es/2010/2/9/13066/comenzando-con-jquery--tutorial-basico-para-comenzar-a-trabajar-con-jquery--parte-1-de-4-

http://www.maestrosdelweb.com/editorial/javascript-facil-y-rapido-con-jquery/

http://www.mkyong.com/jquery/

http://www.noupe.com/jquery/50-amazing-jquery-examples-part1.html

http://www.masquewordpress.com/240-plugins-para-jquery-lo-mejor-de-lo-mejor/#

http://jquerybyexample.blogspot.com/