jQuery

INTRODUCCIÓN

jQuery es una libreria de Javascript

El propósito de jQuery es hacer que JavaScript sea mucho más fácil usar.

INTRODUCCIÓN

CARACTERÍSTICAS:

- ✔ jQuery es una librería de JavaScript.
- ✔ Simplifica en gran medida la programación JavaScript.
- ✔ Es fácil de aprender.

NOS PROPORCIONA:

- ✔ Manipulación HTML/DOM.
- ✔ Manipulación CSS.
- ✔ Métodos de eventos HTML.
- ✔ Efectos y animaciones.
- ✔ AJAX.
- ✔ Entre otros.

AÑADIR jQuery

Para añadir jQuery, podemos:

- ✔ Descargar la libreria desde http://jquery.com/

La librería jQuery es un solo archivo JavaScript y lo referenciamos con la etiqueta HTML <script> (debe estar dentro de la sección <head>):

```
<head>
<script src="jquery-3.2.1.min.js"></script>
</head>
```

AÑADIR jQuery

Para añadir jQuery, podemos:

- ✔ Incluir jQuery de un CDN (Content Delivery Network):

Google CDN:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
</head>
```

Microsoft CDN:

```
<head>
<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.2.1.min.js"></script>
</head>
```

SINTAXIS

Con jQuery se seleccionan (consultan/query) elementos HTML y se realizan "acciones" sobre ellos.

SINTAXIS BÁSICA:

$(selector).action()

- ✔ Un signo **$** para definir/acceder a jQuery.
- ✔ Un **(selector)** para “consultar/query (o buscar)" elementos HTML.
- ✔ Una **acción()** jQuery que se realizará en el elemento.

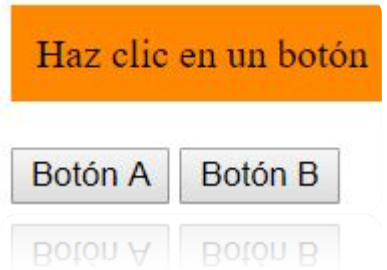
SINTAXIS

EJEMPLOS:

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

EJEMPLO

```
<div id="capa" style="padding: 10px; background-color: #ff8800">Haz clic en un botón</div>

<input type="button" value="Botón A" onclick="$('#capa').html('Has hecho clic en el botón <b>A</b>')">
<input type="button" value="Botón B" onclick="$('#capa').html('Recibido un clic en el botón <b>B</b>')">
```



¿Cómo resolveríamos este mismo ejemplo sin utilizar jQuery?

DOCUMENT READY EVENT

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

   // jQuery methods go here...

});
```

De esta manera, evitamos que cualquier código jQuery se ejecute antes de que el documento termine de cargar (***is ready***).

Se debe tener presente que algunas acciones podrían fallar si se ejecutan antes de que el documento este completamente cargado:

- ✔ Intentar ocultar un elemento que aún no se ha creado.
- ✔ Intentar obtener el tamaño de una imagen que aún no ha sido cargada.

DOCUMENT READY EVENT

Existe incluso una manera más simplificada de hacer referencia a lo mismo:

```
$(function(){

   // jQuery methods go here...

});
```

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

   // jQuery methods go here...

});
```

=

```
$(function(){

   // jQuery methods go here...

});
```

SELECTORES

Los selectores jQuery permiten seleccionar y manipular elementos HTML.

Los selectores jQuery se utilizan para "buscar" (o seleccionar) elementos HTML basados en su:

- ✔ nombre,
- ✔ id,
- ✔ clases,
- ✔ tipos,
- ✔ atributos,
- ✔ valores de atributos
- ✔ y mucho más.

Se basa en los selectores CSS existentes y, además, tiene algunos selectores personalizados.

SELECTORES

Los selectores jQuery permiten seleccionar y manipular elementos HTML.

En: http://api.jquery.com/category/selectors/ se puede encontrar una completa referencia sobre los selectores de la biblioteca.

SINTAXIS:

$('selector')

Los selectores jQuery retornan un array de objetos que coinciden con los criterios especificados. Este array es un conjunto de objetos de jQuery.

SELECTORES

Selector por nombre de etiquetas:

Simplemente indicamos la etiqueta a la que deseamos referirnos, es decir, la etiqueta que queremos seleccionar. Obtendremos así, todas las etiquetas de la página indicada en el selector.

EJEMPLO:

- ✔ **$("h1"):** selecciona todos los encabezados de nivel 1, es decir, todos los tags h1.

SELECTORES

Selector por identificador:

Sirven para seleccionar los elementos que tengan un identificador dado, que se asigna a las etiquetas a través del atributo id del HTML. Para utilizar este selector se indica primero el carácter "#" y luego el identificador de cuyo elemento se desee seleccionar, tal como con CSS.

EJEMPLO:

- ✔ **$("#idElemento")**: selecciona una etiqueta que tiene el atributo id="idElemento"

SELECTORES

Selector por clase:

Podemos indicar el nombre de una clase (class de CSS) y seleccionar todos los elementos a los que se ha aplicado esta clase. Para ello, como en CSS, comenzamos colocando el carácter "." y luego el nombre de la clase que deseamos seleccionar.

EJEMPLO:

- ✔ **$(".miClase"):** selecciona todos los elementos que tienen el atributo class="miClase"

SELECTORES

Selector por varias clases:

Si lo deseamos, podemos indicar varias clases CSS, para obtener todos los elementos que tienen esas clases aplicadas: todas al mismo tiempo. Esto se consigue comenzando por un ".", igual que los selectores de clases, y luego otro "." para separar las distintas clases que queremos utilizar en el selector.

EJEMPLO:

- ✔ **$(".clase1.clase2"):** selecciona los elementos que tienen class="clase1 clase2”.

SELECTORES

Selector asterisco "*":

Nos sirve para seleccionar todos los elementos de la página.

EJEMPLO:

- ✔ **$("*"):** selecciona todos los elementos que tiene la página.

SELECTORES

MÁS SELECTORES:

- ✔ **etiqueta.clase:** Encuentra elementos del tipo de la etiqueta que tengan la clase “clase”.

  Ejemplo: **$(‘div.miClase’);**

- ✔ **selector1, selector2,…:** Encuentra todos los todos los selectores especificados. Equivalente al OR.

  Ejemplo: **$(‘#id1, #id2’);**

- ✔ Muchos más…

EACH

Se trata de un método para realizar acciones con todos los elementos que concuerdan con una selección realizada con la función $(‘selector).

Es muy útil, ya que nos da una manera cómoda de iterar con elementos de la página y hacer cosas con ellos de una manera rápida.

¿CÓMO FUNCIONA?

EACH

¿CÓMO FUNCIONA?

- ✔ Con each realizamos una iteración por todos los elementos del DOM que se hayan seleccionado.
- ✔ Con la variable "this" tenemos acceso al elemento actual.
- ✔ El método each recibe una función que es la que se tiene que ejecutar para cada elemento y dentro de esa función con la variable "this" tenemos una referencia a ese elemento del DOM. Adicionalmente, la función que se envía a each, puede recibir un parámetro que es el índice actual sobre el que se está iterando.

EACH

EJEMPLO:

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

Imaginemos que tenemos una serie de párrafos en la página y queremos cambiar el color de fondo a los mismos, de manera que tengan colores alternos, para hacer dinámicamente un típico diseño para los listados.

Entonces podríamos hacer lo siguiente:

```
$("p").each(function(i){
   if(i%2==0){
      $(this).css("background-color", "#eee");
   }else{
      $(this).css("background-color", "#ccc");
   }
});
```

EACH

EJEMPLO COMPLETO:

```
<html>
<head>
   <title>each del core de jQuery</title>
   <script src="../jquery-1.3.2.min.js" type="text/javascript"></script>
<script>
$(document).ready(function(){
   $("p").each(function(i){
      if(i%2==0){
         $(this).css("background-color", "#eee");
      }else{
         $(this).css("background-color", "#ccc");
      }
   });
});
</script>
</head>
<body>
<p>Primer párrafo</p>
<p>Otro párrafo</p>
<p>Tecer párrafo</p>
<p>Uno más</p>
<p>y acabo...</p>
</body>
</html>
```

EVENTOS

Recordemos el concepto de eventos dado cuando vimos JavaScript…

Ejemplos:

- ✔ mover el mouse sobre un elemento,
- ✔ clic en un botón,
- ✔ clic en un elemento,
- ✔ etc.

En **jQuery**, tenemos un método equivalente para el manejo de eventos que vimos con JS.

Por ejemplo: para asignar un evento al hacer clic a todos los tags “p” de una página

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

EVENTOS

En **jQuery**, tenemos un método equivalente para el manejo de eventos que vimos con JS.

Por ejemplo: para asignar un evento al hacer clic a todos los tags “p” de una página

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

Pero… nos falta definir que acciones deberían ejecutarse cuando el evento es **lanzado**:

```
$("p").click(function(){
  // action goes here!!
});
```

EVENTOS

MÁS UTILIZADOS:

- ✔ **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.
- ✔ **mouseenter():** Este evento se produce al situar el mouse encima de un elemento de la página.
- ✔ **mouseleave():** Este se desata cuando el mouse sale de encima de un elemento de la página.

EVENTOS

MÁS UTILIZADOS:

- ✔ **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 mouse.
- ✔ **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.
- ✔ **focus():** La función se ejecuta cuando el campo de un formulario obtiene el foco.

EVENTOS

MÁS UTILIZADOS:

- ✔ **keydown():** Este evento se produce en el momento que se presiona una tecla del teclado, independientemente de si se libera la presión o se mantiene. Se produce una única vez en el momento exacto de la presión.
- ✔ **keypress():** Este evento ocurre cuando se digita un carácter, o se presiona otro tipo de tecla. Se ejecuta 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, es decir, al dejar de presionar una tecla que teníamos pulsada.

EVENTOS

EVENTO ON:

El método **on()** asigna uno o más manejadores de eventos para los elementos seleccionados.

Ejemplos:

```
$("p").on("click", function(){
    $(this).hide();
});
```

```
$("p").on({
    mouseenter: function(){
        $(this).css("background-color", "lightgray");
    },
    mouseleave: function(){
        $(this).css("background-color", "lightblue");
    },
    click: function(){
        $(this).css("background-color", "yellow");
    }
});
```

MANIPULACIÓN DEL DOM

jQuery contiene potentes métodos para cambiar y manipular **elementos y atributos HTML**.

En cuanto a los **elementos**, podemos mencionar tres métodos que son muy simples. Estos son:

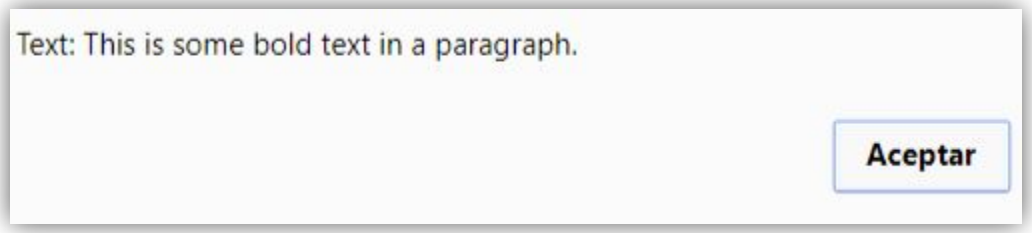
- ✔ **text()** - Establece o devuelve el **contenido de texto** de los elementos seleccionados
- ✔ **html()** - Establece o devuelve el **contenido** de los elementos seleccionados.
- ✔ **val()** - Establece o devuelve el **valor** de los campos de formulario

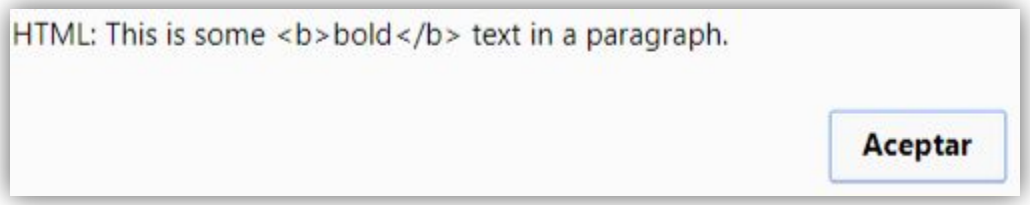
MANIPULACIÓN DEL DOM

EJEMPLO GET(OBTENER)

DIFERENCIA ENTRE TEXT() Y HTML():

- ✔ **text()** - Establece o devuelve el contenido de texto de los elementos seleccionados
- ✔ **html()** - Establece o devuelve el contenido de los elementos seleccionados.





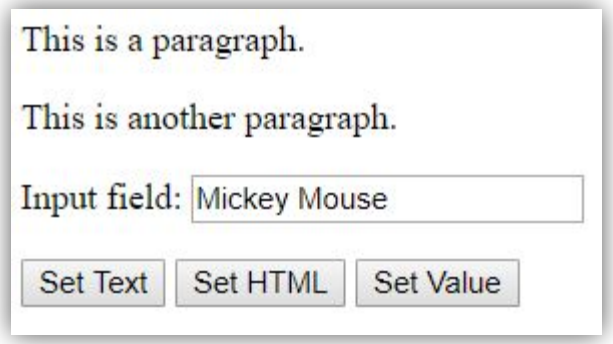

```
<script>
$(document).ready(function(){
    $("#btn1").click(function(){
        alert("Text: " + $("#test").text());
    });
    $("#btn2").click(function(){
        alert("HTML: " + $("#test").html());
    });
});
</script>
</head>
<body>

<p id="test">This is some <b>bold</b> text in a paragraph.</p>

<button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>
```

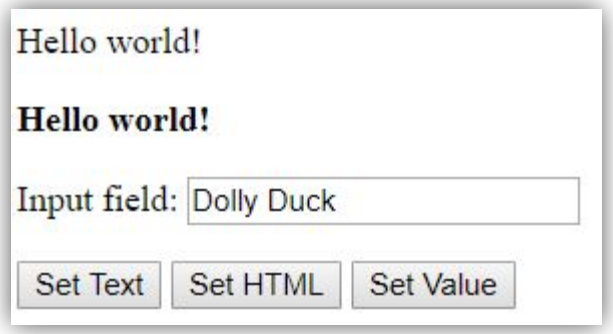
This is some **bold** text in a paragraph.

Show Text | Show HTML

MANIPULACIÓN DEL DOM

EJEMPLO SET(ESTABLECER)






```
.script>
$(document).ready(function(){
    $("#btn1").click(function(){
        $("#test1").text("Hello world!");
    });
    $("#btn2").click(function(){
        $("#test2").html("<b>Hello world!</b>");
    });
    $("#btn3").click(function(){
        $("#test3").val("Dolly Duck");
    });
});
</script>
</head>
<body>

<p id="test1">This is a paragraph.</p>
<p id="test2">This is another paragraph.</p>

<p>Input field: <input type="text" id="test3" value="Mickey Mouse"></p>

<button id="btn1">Set Text</button>
<button id="btn2">Set HTML</button>
<button id="btn3">Set Value</button>
```

MANIPULACIÓN DEL DOM

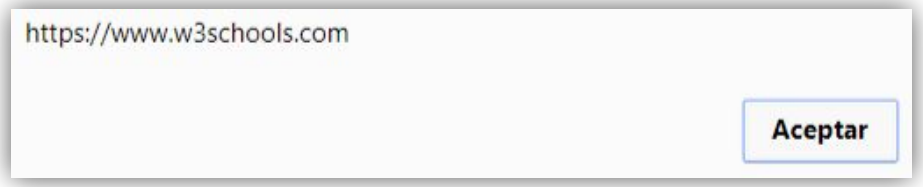
En cuanto a los **atributos**, tenemos el método **attr()**. Pero antes, recordemos la siguiente sintaxis:

Entonces:

- ✔ **attr():** se utiliza para obtener valores de atributos.

MANIPULACIÓN DEL DOM

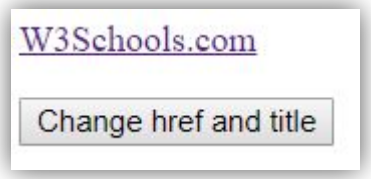
EJEMPLO GET(OBTENER)

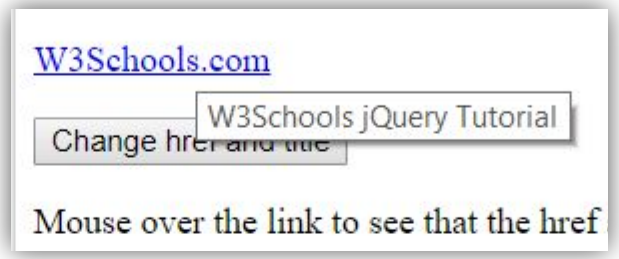



```
<script>
$(document).ready(function(){
    $("button").click(function(){
        alert($("#w3s").attr("href"));
    });
});
</script>
</head>
<body>

<p><a href="https://www.w3schools.com" id="w3s">W3Schools.com</a></p>

<button>Show href Value</button>

</body>
</html>
```

MANIPULACIÓN DEL DOM

EJEMPLO SET(ESTABLECER)






```
.script>
$(document).ready(function(){
    $("button").click(function(){
        $("#w3s").attr({
            "href" : "https://www.w3schools.com/jquery",
            "title" : "W3Schools jQuery Tutorial"
        });
    });
});
</script>
</head>
<body>

<p><a href="https://www.w3schools.com" title="some title" id="w3s">W3Schools.com</a></p>

<button>Change href and title</button>

<p>Mouse over the link to see that the href attribute has changed and a title attribute is set.</p>
```

MANIPULACIÓN DEL DOM

AGREGAR ELEMENTOS/CONTENIDO:

Veremos cuatro métodos jQuery que se utilizan para agregar contenido nuevo:

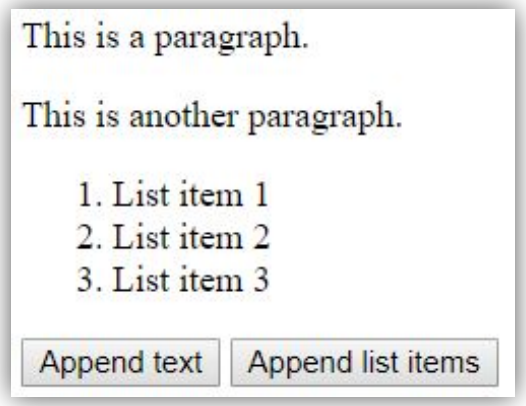
- ✔ **append()** - Inserta contenido al final de los elementos seleccionados
- ✔ **prepend()** - Inserta contenido al principio de los elementos seleccionados
- ✔ **after()** - Inserta contenido después de los elementos seleccionados
- ✔ **before()** - Inserta contenido antes de los elementos seleccionados

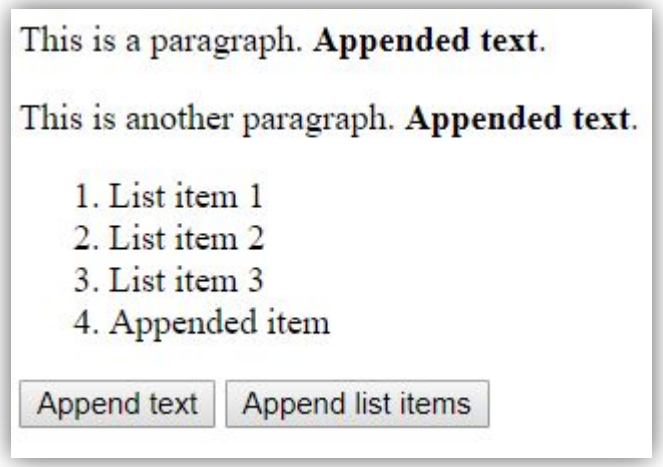
TIP: A estos métodos le podemos pasar más de un parámetro para agregar más de un elemento.

MANIPULACIÓN DEL DOM

EJEMPLO:

```
<script>
$(document).ready(function(){
    $("#btn1").click(function(){
        $("p").append(" <b>Appended text</b>.");
    });

    $("#btn2").click(function(){
        $("ol").append("<li>Appended item</li>");
    });
});
</script>
</head>
<body>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<ol>
  <li>List item 1</li>
  <li>List item 2</li>
  <li>List item 3</li>
</ol>

<button id="btn1">Append text</button>
<button id="btn2">Append list items</button>
```





MANIPULACIÓN DEL DOM

ELIMINAR ELEMENTOS/CONTENIDO:

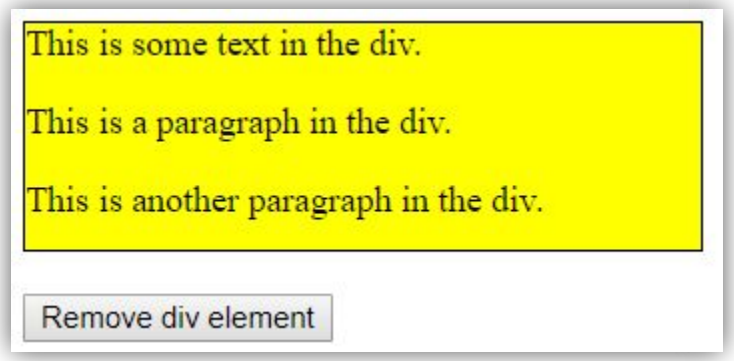
Para eliminar elementos y contenido, hay principalmente dos métodos jQuery:

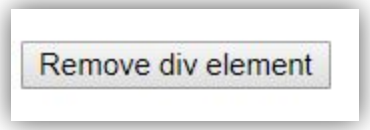
- ✔ **remove()** - Elimina el elemento seleccionado (y sus elementos secundarios)
- ✔ **empty()** - Elimina los elementos secundarios del elemento seleccionado

MANIPULACIÓN DEL DOM

EJEMPLO REMOVE:

```
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").remove();
    });
});
</script>
</head>
<body>

<div id="div1" style="height:100px;width:300px;border:1px solid black;background-color:yellow;">

This is some text in the div.
<p>This is a paragraph in the div.</p>
<p>This is another paragraph in the div.</p>

</div>
<br>

<button>Remove div element</button>
```

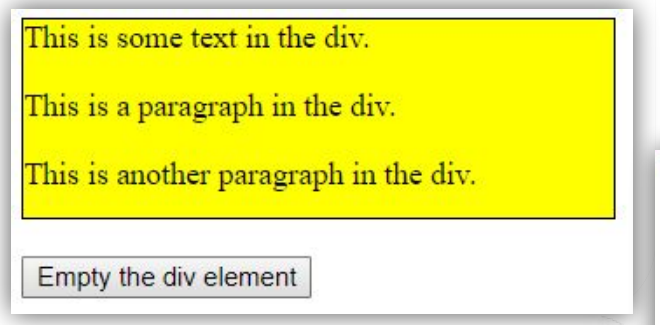




MANIPULACIÓN DEL DOM

EJEMPLO REMOVE:

```
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").empty();
    });
});
</script>
</head>
<body>

<div id="div1" style="height:100px;width:300px;border:1px solid black;background-color:yellow;">

This is some text in the div.
<p>This is a paragraph in the div.</p>
<p>This is another paragraph in the div.</p>

</div>
<br>

<button>Empty the div element</button>
```



MANIPULACIÓN DE ESTILOS CSS

jQuery, provee una fácil manipulación de los estilos CSS de los elementos.

Para esto, tiene los siguientes métodos:

- ✔ **addClass()** - Agrega una o más clases a los elementos seleccionados.
- ✔ **removeClass()** - Elimina una o más clases de los elementos seleccionados.
- ✔ **toggleClass()** - Alterna entre añadir/quitar clases de los elementos seleccionados.
- ✔ **css()** - Establece o devuelve el atributo de estilo.

MANIPULACIÓN DE ESTILOS CSS

MÉTODO CSS():

El método **css()** establece o retorna una o más propiedades de estilo para los elementos seleccionados.

SINTAXIS PARA RETORNAR UNA PROPIEDAD:

```
css("propertyname");
```

EJEMPLO:

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

MANIPULACIÓN DE ESTILOS CSS

MÉTODO CSS():

El método **css()** establece o retorna una o más propiedades de estilo para los elementos seleccionados.

SINTAXIS PARA ESTABLECER UNA PROPIEDAD:

```
css("propertyname","value");
```

EJEMPLO:

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

MANIPULACIÓN DE ESTILOS CSS

MÉTODO CSS():

El método **css()** establece o retorna una o más propiedades de estilo para los elementos seleccionados.

SINTAXIS PARA ESTABLECER MÚLTIPLES PROPIEDADES:

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

EJEMPLO:

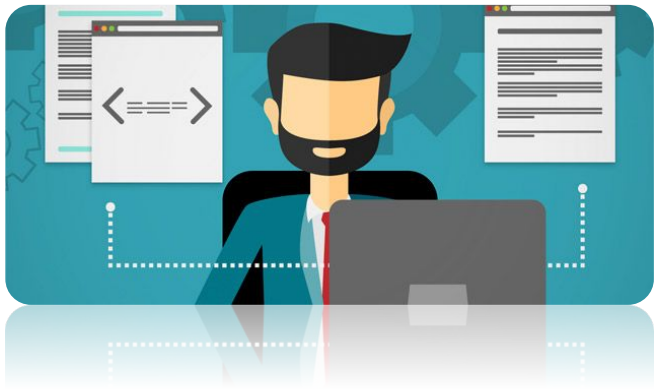
```
$("p").css({"background-color": "yellow", "font-size": "200%"});
```

EJERCICIOS – PARTE I

EJERCICIO 1: Mostrar el texto “READY!” una vez que el documento HTML ha terminado de cargar.

EJERCICIO 2: Crear un botón con el texto “Mostrar contenido oculto” y mediante el uso de eventos, hacer que al hacer clic sobre el mismo se muestre debajo un texto a elección.

EJERCICIO 3: Crear un párrafo con algún texto, y agregarlo al final del body una vez que el documento ha sido cargado en su totalidad (este contiene al menos dos <p>).



EJERCICIOS – PARTE II

EJERCICIO 4: Generar un HTML que contenga 4 párrafos de texto, luego, usando jQuery obtener todos los tags <p> y recorrerlos, asignándoles a cada uno un color de texto diferente.

TIP: Para recorrerlos usar EACH.

EJERCICIO 6: Crear un check-box (el mismo podría ser para representar los términos y condiciones de una página web) y por defecto, dejar el botón submit desactivado. Al chequearlo, habilitar el botón para dejar al usuario hacer dicho submit.

Ejercicio

RESOLVER CON JQUERY

Resolver un sistema de valoración con 5 estrellas, con las siguientes pautas:

Se mostrará el valor actual al inicio. 

Para valorar, al pasar con el mouse se iluminarán.

Indicar que se ha valorado y cambiar el valor si corresponde