jQuery

Bootstrap es a CSS como jQuery es a Javascript. En otras palabras, jQuery es una **librería** que agrupa muchas **funciones javascript** para **facilitar nuestro flujo de trabajo y simplificar la interacción**.

La librería jQuery simplifica la tarea de manipular una página HTML luego de que ha sido desplegada en el navegador que permitirán al usuario interactuar con la página. Además posee características que ayudan en la creación de animaciones y DHTML (No recargar la página para actualizar información mediante la manipulación del DOM) y la comunicación de forma asíncrona con el servidor (Ajax).

No hay que olvidar que el script sigue siendo de lenguaje `Javascript`

¿Por qué jQuery?

Existen muchas bibliotecas de `javascript` pero jQuery es uno de los **más populares** y por ende uno de los **más extensibles**.

Grandes compañías usan jQuery, como por ejemplo:

- Google
- Microsoft
- IBM
- Netflix

Principales características

- Licencia MIT y Open Source. Excelente soporte, **actualización constante** y **comunidad activa**.

- Compatibilidad con todos los navegadores web.
- Accede al documento HTML (DOM = Document Object Model)
- **Manipulación de estilos CSS** (Compatibilidad con CSS3)
- Excelente **integración con AJAX.**
- Creación de **animaciones** y **efectos visuales.**
- Manejo de **eventos** para los elementos de la página web.
- **Simplifica** tareas o rutinas comunes de Javascript.
- Manipulación JavaScript Object Notation (JSON).
- Su peso es muy **reducido**.
- **Ahorro significativo** en el tiempo de creación de una web.

¿Como incorporar JQuery a nuestro proyecto?

De forma remota:

Para una instalación remota debemos agregar **UNO** de los siguientes scripts a nuestra página HTML

CDN (Content Delivery Network), es que una red de servidores web conectados entre sí, dispersos geográficamente, que almacenan el contenido de un sitio para entregarlo de forma más eficiente

Desde el CDN de Google:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
```

Desde el CDN de Microsoft

```
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.2.2.js"></script>
```

De forma local:

Descargar jQuery desde https://jquery.com/.

Luego se copia el archivo descargado a la carpeta `js` de nuestro proyecto web y se agrega al documento HTML como sigue:

```
<script src="/js/jquery-2.2.2.js"></script>
```

Primeros pasos

Sintáxis básica

jQuery está diseñado especialmente para seleccionar un elemento del documento HTML y realizar una acción sobre el o responder a un evento. La sintáxis básica es la siguiente:

Si responde a una acción: `$(selector).accion()`

Si responde a un evento: `$(selector).evento()`

Evento Ready

Ready es un evento que utilizaremos para poder trabajar sobre el DOM, permitiendo que este se ejecute sin problemas. Lo que hace es esperar hasta que todos los elementos de la página se carguen correctamente, evitando que `Javascript` intente usar elementos que aún no existan en el documento. Su implementación es la siguiente (de tal modo también es posible utilizar JQuery sin usar este evento, pero, es recomendable a la hora de asignar callbacks a los eventos cuando el DOM ya este completamente cargado):

```
<script>

$(document).ready(function(){

   // código a ejecutar

});

</script>
```

Selectores

Los selectores son una de las funciones más útiles de jQuery, permiten seleccionar y manipular elementos del documento HTML.

jQuery nos permitirá buscar elementos basándose ya sea en su nombre, id, clase, tipo, atributo, etc. Se basa convenientemente en los selectores de CSS y además posee algunos selectores propios.

Todos los selectores parten con el signo peso y un paréntesis `$()`

Ejemplos

```
$("p") selecciona todos los elementos <p>.

$(".test") selecciona todos los elementos de la clase "test".

$("#test") selecciona el elemento con la id  "test".

$(this) selecciona el elemento actual
```

Eventos

Hasta el momento podemos ver que JQuery nos permitirá manipular el DOM simplificando el código javascript, mediante las diversas funciones que contiene esta biblioteca de javascript.

Los eventos, se basan en esperar que el usuario realice una acción sobre cierto elemento, ¿Cómo saber cuándo el usuario interactúa? Mediante "escuchadores" (listeners).

Existen muchos eventos, sin embargo sólo se hará referencia a los dos más utilizados, puede revisar el resto en Eventos jQuery.

Otros eventos: `blur`, `focus`, `keydown`, `load`, `mousemove`, `resize`, `scroll`, `submit`, `select`.

```
$('element').click()
```

Como su nombre indica, este listener se ejecutará cuando hagamos click en el elemento que hayamos seleccionado, se utiliza como sigue:

```
$(document).ready(function(){
    $( "#elemento" ).click(function() {
        alert( "Evento click!!!" );
    });
}
```

```
$('element').hover()
```

Este evento se desencadenará cuando detecte que el mouse pasa por sobre el elemento.

Se usa generalmente en botones e imágenes, cambiando alguna propiedad de estos, haciendo intuir al usuario que es un elemento interactivo.


```
$(document).ready(function(){
    $( "#elemento" ).hover(function() {
        alert( "Evento hover!!!" );
    });
}
```

Efectos

fadeIn() / fadeOut()

Este efecto hace aparecer o desaparecer un elemento en el DOM desvaneciéndose en un tiempo determinado.

Para usarlo simplemente lo llamamos después de elegir un elemento, en este caso al hacer click sobre un botón se desvanecerán o aparecerán todos los elementos de tipo p

```
$(document).ready(function(){
    $('#boton').click(function() {
        $('p').fadeOut();
    });

    $('#boton2').click(function() {
        $('p').fadeIn();
    });
}
```

toggle()

Toggle nos permite comprobar el estado actual del elemento (ya sea en estado fadeIn o fadeOut) y cambiarlo por su contraparte, en este caso necesitamos solo un botón para ejecutar las funciones de fadeIn o fadeOut según corresponda.

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

slideUp() / slideDown()

Este efecto nos permite hacer aparecer o desaparecer elementos como si se estuviesen "deslizando".

Menos palabras, más acción. La forma de usarlo es igual a cualquier otro efecto.

```
$("#flip").click(function(){
    $("#panel").slideDown();
});

$("#flip").click(function(){
    $("#panel").slideUp();
});
```

slideToggle()

Al igual que `toggle()` nos permite cambiar de estado entre slideUp() y slideDown() según corresponda.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#flip").click(function(){
        $("#panel").slideToggle("slow");
    });
});


</script>

<style>
#panel, #flip {
    padding: 5px;
    text-align: center;
    background-color: #e5eecc;
    border: solid 1px #c3c3c3;
}

#panel {
    padding: 50px;
    display: none;
}
</style>
</head>
<body>
<div id="flip">Click to slide the panel down or up</div>
<div id="panel">Hello world!</div>

</body>
</html>
```

Timing

Para cualquiera de los efectos podemos definir qué tan rápido queremos que ocurra, para esto usamos la siguiente sintaxis:

```
$("elemento").efecto(tiempo);
```

jQuery cuenta con algunos tiempos predefinidos los cuales son `slow`, `fast` o también podemos definirlo en milisegundos

```
$("elemento").efecto("slow");
$("elemento").efecto("fast");
$("elemento").efecto(3000);
```

Podemos incluso desencadenar acciones luego de que un efecto termine de ejecutarse, para esto:

```
$("#elemento").fadeOut("slow", function() {
    // hacer algo cuando el elemento termine de desaparecer
}
```

HTML

GET

Podemos **obtener** valores o contenido desde el documento HTML con tres simples funciones:

- **text()** - Devuelve el contenido de texto del elemento seleccionado`

```
var texto = $("#texto").text();
alert(texto);
```

- **html()** - Devuelve el contenido de un elemento incluyendo las etiquetas html

```
var contenido = $("#container").html();
alert(contenido);
```

- **val()** - Devuelve el valor contenido en algún campo de formulario.

```
<input type="text" id="test" value="Valor ingresado">

...

var valor = $("#test").val();
alert(valor);
```

SET

Para **cambiar** el contenido de los elementos se utilizan las mismas funciones

- text()

```
$("#elemento").text("Texto nuevo.");
```

- html()

```
$("#elemento").html("<p>Contenido nuevo.</p>");
```

- val()

```
$("#test").val("Valor nuevo");
```

ADD

Añade nuevo contenido HTML Hay cuatro métodos que nos permitirán insertar contenido relativo a nuestro elemento seleccionado

- **append()** - Inserta contenido al final de los elementos seleccionados

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

...

    $("ol").click(function(){
        $(this).append("<li>Elemento nuevo de lista</li>");
    });
```

- **prepend()** - Inserta contenido al principio de los elementos seleccionados

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

...

    $("ol").click(function(){
        $(this).prepend("<li>Elemento nuevo de lista</li>");
    });
```

CSS

jQuery has several methods for CSS manipulation. We will look at the following methods:

- **addClass()** - Agrega una clase al elemento seleccionado, la clase debe estar previamente definida en nuestro css

```
/* archivo .CSS */

.blue {
    color: blue;
}


/* archivo .HTML */

...

$("#button").click(function(){
    $("h1, h2, p").addClass("blue");
});
```

- **removeClass()** - Quita una o más clases del elemento

```
$("#button2").click(function(){
    $("h1, h2, p").removeClass("blue");
});
```

- **toggleClass()** - Agrega o quita una clase determinada según corresponda

```
$("#button").click(function(){
    $("h1, h2, p").toggleClass("blue");
});
```

Actividad 1

Cree un documento HTML que contenga un elemento que cambie cuando el mouse pase por sobre el.*Puede ser un botón, imagen, div, etc.

Método css()

- **Return CSS**: Obtiene el valor de la clase css que consultemos. Arroja el valor del PRIMER elemento que encuentre el selector.

```
alert( "Tamaño de fuente h1: " +  $("h1").css("font-size") );
```

- **Set CSS**

Inserta una característica CSS en un elemento. La propiedad se agrega a todos los elementos que cumplan con el selector.

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

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

Plugins jQuery

Estos son algunos ejemplos de la gran cantidad de plugins disponibles para jQuery.

- ToolTipster
- Knob
- Elevator
- avgrund
- Justified
- DataTables
- FullPage

Actividad 2

Utilizar uno de los anteriores plugins JQuery. Puede basarse en alguna tarea anterior.

Actividad 3

Se pedirá elaborar un nuevo proyecto web, donde deberá crear una vista de un chat mediante el uso de Bootstrap y JQuery, puede tomar como referencia para sus css aplicaciones como Facebook Messenger, Whatsapp, Telegram, Hangouts.

Dentro de la funcionalidad de la aplicación, se debe escribir un texto en un campo input, una vez que se envíe el mensaje, dicho campo debe quedar vacío, además, el mensaje debe aparecer en la parte superior del campo de entrada de texto, este debe incorporar efectos mediante JQuery.