CLIENT SIDE WEB DEVELOPMENT
TEMA6: JQUERY RESUME

CFGS - DAW 2

JESUITAS

Sagrado Corazón
Jesuitas · Logroño

Index

1. Uso de jquery

- Para poder usar jquery, es necesario incorporar a nuestro marco de trabajo la librería de jQuery.
  - Descargar archivo.js e incorporarlo dentro de la etiqueta <script>

```
<script type="text/JavaScript" src="libreria-jQuery.js">
```

  - Hacer uso del CDN (Content Delivery Network)

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
```

2. Uso JQuery

- El uso de la librería jQuery y de sus funciones se realiza a través de la función jQuery(selector) o su equivalente mediante el uso de $,
  - $(selector)
  - Dónde selector es una cadena de texto con la descripción de una regla basada en CSS.

1. Ejemplos de funcionamiento

- Detección de la carga del DOM

```
$(document).ready (miFuncion()
 {//codigo
 }
 );
```

- Concatenación de instrucciones
  - De tal forma, que en una única línea de código, podemos realizar diferentes operaciones aprovechando los objetos jQuery devueltos por cada función.

1. Ejemplos de funcionamiento

- Encadenamiento

La gran mayoría de los diferentes métodos definidos en jQuery devuelven a su vez otro objeto jQuery. Este tipo de operaciones es lo que llamamos encadenamiento, cada operación se separa de la siguiente mediante el símbolo '.'

```
$("a").click(miFuncion).addClass("nuevaClase");
```

En cada elemento anchor <a> del documento, definimos un manejador para el evento click y aplicamos las reglas de estilo añadiendo una nueva clase.

1. Ejemplos de funcionamiento

- En javaScript:
  - La implementación de esta funcionalidad requeriría al menos:
    - Una estructura de control (for, while) que permitiera manipular una colección de elementos de tipo anchor: getElementByTagName()
- En jQuery:

```
$("a").click(miFuncion).addClass("nuevaClase");
```

1. Ejemplos de funcionamiento

- En jQuery:

Ejemplo1.html

```
<script>
    $(document).ready(function(){
        $("div.opciones")
            .find("li")
            .eq(1)
            .append("nuevo")
            .end()
            .eq(0)
            .append("antiguo")
        }
    );

</script>
```

- Al manejar encadenamientos, no es necesario todo el código en la misma línea.
- Si en algún punto de la cadena, queremos recuperar el contenido del objeto jQuery previo a la llamada de un método, bastará con hacer uso de la llamada al método end().

2. Selección

- Permite realizar la selección de nodos mediante el uso del selector $ y mediante métodos específicos incluidos dentro de la propia librería accesibles a través del objeto jQuery.
- Estos métodos se pueden utilizar de forma separada o combinados a la hora de realizar una selección.
- Selectores básicos:

DOM	JQuery
Document.getElementById("txtdatos")	$("#txtdatos")
getElementsByTagName('p')	$('p')
getElementsByClassName('clase')	$('.clase')
	$('*'): se aplica a todos los elementos de la página.

2. Selección. Selectores jerárquicos.

- Selectores con modificadores jerárquicos: Ejemplo 2

```
<script>
 //Cambiamos el texto de los nodos <p>
 //hijos del nodo cuyo atributo id es #datos
  $(function()
     {
       $("#datos > p").text("Texto cambiado");
     })
  //Selector combiando Css con expresiones navegación
  function funcionXPath()
  {
     $("#datos > p:first-child").text("Soy primero");
     $("#datos > p:first-child+p").text("Soy segundo");
     $("#datos > p:first-child+p+p").text("Soy tercero");
     }

</script>
```

2. Selección. Selectores jerárquicos.

- Tabla de selectores jerárquicos

Selector	Descripción
E F	Elementos F hijos de E no necesariamente de primer nivel.
E > F	Elementos F hijos de E de primer nivel.
F	
E + F	Elementos de tipo F justo despúes de uno de tipo E
E,F	Elementos que coincidan con cualquiera de los selectores E o F
E ~ F	Elementos de tipo F que tienen como predecesor un elemento de tipo E en el mismo nivel sin importar si son o no adyacentes.

2. Selección. Selectores por búsqueda.

Selector	Descripción
E[att]	Elementos cuyo selector corresponde con el elemento E y además contiene el atributo att
E[att=valor]	Elementos cuyo selector corresponde con el elemento E y además contiene el atributo att con el valor especificado.
E[att*=valor]	Elementos cuyo selector corresponde con el elemento E y además contiene el atributo att cuyo valor contiene la cadena especificada.

2. Selección. Selectores para identificar y seleccionar elementos dentro de un formulario.

Selector	Descripción
:input	Selecciona elementos de tipo input, select, textarea, button
:text	Selecciona elementos del tipo input[type=text]
:button	Selecciona elementos del tipo input[type=submit], input[type=button], input[type=reset] y button.
:file : image **:password**	Selecciona elementos del tipo input type=, entre los que tenemos input[type=file], input[type=image], input[type=password], input[type=reset], input[type=submit]. **:reset :submit**
:hidden **:visible**	Selecciona elementos visibles o invisibles
:enabled	Selecciona elementos con los atributos enabled o disabled de acuerdo al selector utilizado
REFERENCIA COMPLETA DE SELCTORES DE JQUERY: En la API accesible en la url: http://api.jquery.com(category/selectors/ donde podemos	

3. Navegar por una colección

- Para recorrer una colección de elementos y realizar operaciones de acuerdo al contenido de sus nodos internos. En el Dom recibíamos una colección de tipo NodeList donde utilizábamos un índice y una estructura for.

- Con Jquery podemos simplificar dicha operación, con el método each() que nos permite iterar a lo largo de toda la colección sin necesidad de prestar atención a índice alguno además de permitir definir una función para manejar cada objeto iterado, el cual se representara dentro de esta función como this.

```
<script>
//$(this) es el objeto actual de la interacción
//each(function(indice,elemento)
//En cada iteración se ejecuta la función anónima, la cual recibe el índice del elemento del array u objeto de la colección
//además del propio elemento u objeto, lo que permite su acceso y manipulación. En el caso de detectar una condición o
//situación que hace innecesario seguir ejecutando la iteración, podemos interrumpirla mendiante la sentencia return false.
 $(function(){
      $("#datos").find("p").each(function(i){
        $(this).append("Parrafo número" + (i+1));
      });
 }
)
 </script>
```

3. Navegar por una colección
Funciones adicionales en Jquery.

- Para recorrer el DOM, filtrar los elementos de una colección, utilizadas sobre todo en los encadenamientos.

Selector	Descripción
Eq	Selecciona elementos de tipo input, select, textarea, button.
Filter	Devuelve los elementos que cumplen el selector.
not	Devuelve los elementos que no se corresponden con el selector. Inverso de filter.
Next	Devuelve el siguiente elemento hermano del actual
Parent	Devuelve el elemento contenedor del elemento actual
Siblings	Devuelve un jquery con todos los nodos hermanos del elemento, Es posible filtrar los hermanos definiendo a us vez un selector dentro del método siblings (selector).
:first-child	Selecciona el primer descendiente del elemento
:last-child	Selecciona el ultimo elemento descendiente del elemento.

3. Navegar por una colección
Filtros básicos en objetos Jquery.

- Para recorrer el DOM, filtrar los elementos de una colección, utilizadas sobre todo en los encadenamientos.

Filtro	Descripción
Posición	
:first	Selecciona el primer elemento de la colección del objeto Jquery obtenida.
:last	Selecciona el último elemento de la colección del objeto Jquery obtenida.
:eq(índice)	Selecciona el elemento que ocupa la posición índice en la colección del objeto Jquery obtenida.
:gt(índice)	Selecciona el elemento que ocupa una posición mayor que el índice en la colección del objeto Jquery obtenida.

3. Navegar por una colección
Filtros básicos en objetos Jquery.

- Para recorrer el DOM, filtrar los elementos de una colección, utilizadas sobre todo en los encadenamientos.

Filtro	Descripción
:even	Selecciona el elemento que ocupa una posición par en el índice en la colección del objeto Jquery obtenida.
:odd	Selecciona los elementos que ocupan una posición impar en el índice en la colección del objeto Jquery obtenida.
:not(selector)	Selecciona los elementos que no cumplen la selección del selector.

Contenido	
:has(selector)	Selecciona los elementos que contengan al menos un descendiente que cumpla la expresión del selector.
:contains(selector)	Selecciona los elementos que contengan al menos un descendiente, con independencia del nivel de este en cuyo contenido se encuentra el texto buscado.
:empty	Selecciona los elementos vacíos.

3. Navegar por una colección
Ejemplos con toogle

- Prueba toogle 1: Alternar la visibilidad de una respuesta

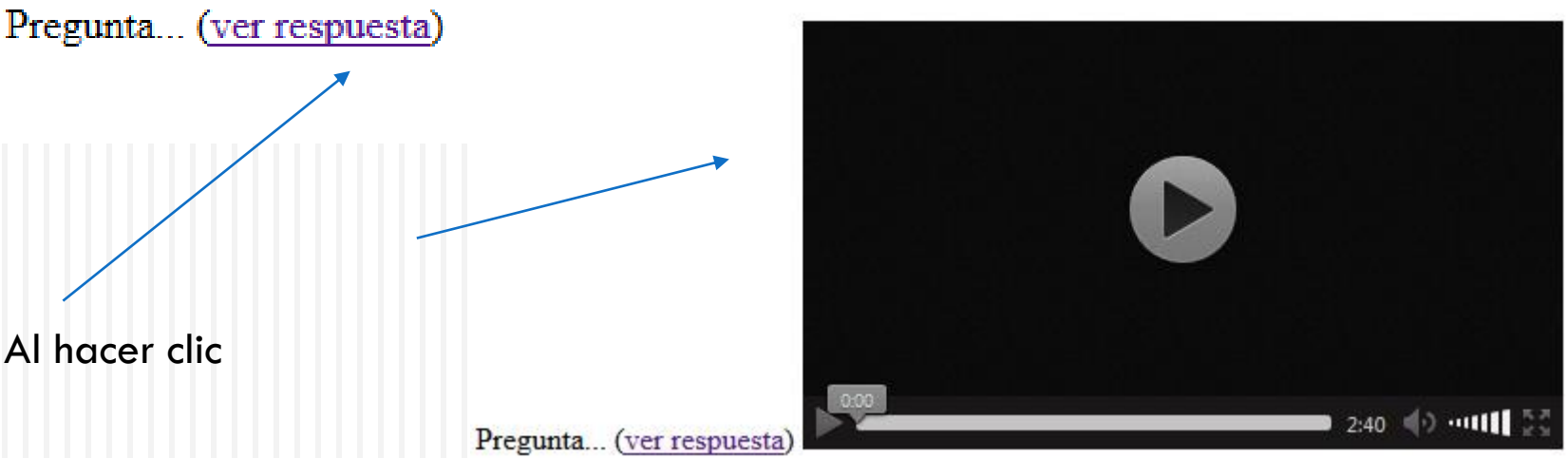
```
<script>
// jQuery
$(document).ready(function(){
    $('#alternar-respuesta-ej1').on('click',function(){
        $('#respuesta-ej1').toggle();
    });
});

</script>
```

- Prueba toogle 2: Animación de un bloque

```
$(document).ready(function(){
    $('#alternar-respuesta-ej2').on('click',function(){
        $('#respuesta-ej2').toggle('slow');
    });
});
```

3. Navegar por una colección
Ejemplos con toogle

- Prueba toogle 3: Animación de un vídeo



- Al hacer clic

- Prueba toogle 4: Abrir y cerrar todas las respuestas a la vez
- Prueba toogle 5: Cambiar el texto de un botón según el click
- Prueba toogle 6: Abrir y cerrar un panel lateral

Carpeta Tema 6 Bis.

4. Inserción de elementos

- Jquery posee diferentes métodos para añadir nuevos elementos al DOM del la página:

Método	Ejemplos
Insertar delante	
Before(contenido)	
insertBefore(contenido)	
Insertar después	
after(contenido)	
insertAfter(contenido)	
Insertar dentro	
append(contenido) // appendTo(contenido)	
prepend(contenido) // prependTo	

5. Añadir manejadores de eventos

- *Desde la versión 1.7 ha aparecido una nueva forma de creación de manejadores para eventos mediante los métodos on y off.*
  $(elementos).on(events[,selector][,data], handler);
- *$(elements).off([events][,selector][,handler]);*
- *Faltan ejemplos:*

5. Añadir manejadores de eventos

- Veamos, las funciones de **¡Query on() y off()** para la asignación y eliminación de eventos a los diferentes elementos del DOM.

```
$(selector).on(evento, childSelector, parametros, funcion, mapeo)

$(selector).off(evento, selector2, funcion, map)
```

```
$(document).ready(function(){
$("#elementocontrolado").on("dblclick
change",function(){
alert('Función on() activada.');
});
});
```

5. Añadir manejadores de eventos

- Veamos, las funciones de **¡Query on() y off()** para la asignación y eliminación de eventos a los diferentes elementos del DOM.

Ejemplo de ¡Query on() para asignar un evento a un elemento del DOM

```
$(document).ready(function(){
$("#parrafo").on("click",function(){
alert('Has hecho click en el párrafo');
});
});
```

•Detectamos el click en el párrafo con id = 'parrafo'.
•Sacamos un alert por pantalla co un mensaje.

```
<p id="parrafo">Haz click en cualquier palabra de este párrafo</p>
```

5. Añadir manejadores de eventos

- Veamos, las funciones de **¡Query on() y off()** para la asignación y eliminación de eventos a los diferentes elementos del DOM.

Ejemplo de ¡Query on() para asignar varios eventos a un elemento del DOM

```
$(document).ready(function(){
$("#elementocontrolado").on("dblclick change",function(){
alert('Función on() activada.');
});
});
```

•Detecto los eventos dblclick (doble click) y change (cambio de valor) en el elemento con id = 'elementocontrolado'.
•Saco un alert con un mensaje por pantalla.

```
<p>Introduce la edad: <input id="elementocontrolado" type="number"></p>
```

5. Añadir manejadores de eventos

- Veamos, las funciones de **¡Query on() y off()** para la asignación y eliminación de eventos a los diferentes elementos del DOM.

Ejemplo de ¡Query on() para el mapeo de eventos a un elemento del DOM

```
$(document).ready(function(){
$("#elementocontrolado").on({
change:function(){alert('El input ha cambiado de valor');},
click:function(){alert('Has hecho click en el input');}
});
});
```

•Mantengo controlado el elemento con id = 'elementocontrolado'.
•Con change (cambio de valor) detecto que el elemento ha variado de valor y saco un mensaje por pantalla
•En segundo lugar detecto el click en el elemento y saco un mensaje por pantalla.

El código HTML del input es idéntico al del ejemplo anterior.

5. Añadir manejadores de eventos

- Veamos, las funciones de **jQuery on() y off()** para la asignación y eliminación de eventos a los diferentes elementos del DOM.
- **Ejemplo de jQuery off() para la eliminación de todos eventos asignados**

```
$("#elementocontrolado").off();
```

- **Ejemplo de jQuery off() para la eliminación de un evento determinado**

```
$("p").off("click",changeSize);
```

6. Modificar propiedades css

- *$(selector).hide() : este método hide() establece la propiedad display:none*
- *Tiene métodos removeClass y addClass*
- *Se pueden realizar todas las operaciones en la misma línea de ejecución de forma encadenada, donde ambos métodos removeClass y addClass se aplicarían sobre el mismo elemento/s.*

```
$("poem li.blue").removeClass("blue").addClass("green");
```

7. Ejemplos: Creación, manipulación e Inyección.

1. Create element and store a reference

```
var p = $('<p>')
```

2. Use a method to manipulate

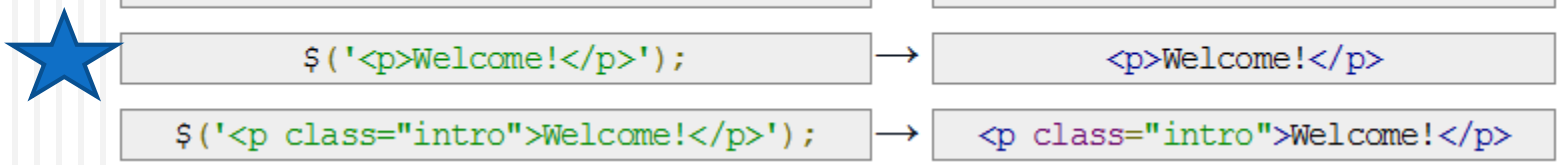
```
p.addClass('special');
```

3. Inject into your HTML

```
$('body').append(p);
```

7. Creación, manipulación e Inyección.

- Create and store:
  - Pass in any HTML string and JQuery will create it and return it as a collection:



  - We can store a reference to our new element in memory:

```
var myParagraph = $('<p class="intro">Welcome!</p>');
```

7. Creación, manipulación e Injección.

- Manipulate:
  - Now that we've stored a reference, we can manipulate it:

```
var myParagraph = $('<p class="intro">Welcome!</p>');
```

```
myParagraph.css('font-size','4em');
```

  - Classes can also be easily added or removed:

```
$('div').addClass('class_name');
$('div').addClass('class_name1 class_name2 class_name3');
```

7. Creación, manipulación e Injección.

- Inject:
  - We can take our stored reference to myParagraph and inject it somewhere:

```
$('body').append(myParagraph);
```

```
$('body').prepend(myParagraph);
```