Tema 6: Integración de contenido interactivo

Ciclo Superior DAW

Asignatura: Diseño de interfaces

Curso 20/21

Introducción

- En este capítulo aprenderemos los siguientes conceptos:
  - Conocer el framework de desarrollo jQuery
  - Conocer los diferentes selectores de jQuery
  - Mostrar los componentes Bootstrap interactivos
  - Conocer las herramientas de personalización y empaquetado de componentes

JQuery

- Es la **librería enriquecida de JavaScript que facilita enormemente el desarrollo de aplicaciones de entorno cliente**

- Dispone de un API de funciones o framework que permite muchas funcionalidades como modificar los estilos de un documento HTML, simplificar la interacción con documentos, manipular el árbol DOM, etc.

JQuery

- JQuery dispone de una extensa documentación, tutoriales, descargas, etc.

- Los podemos descargar de https://jquery.com/

JQuery. Características

- Total compatibilidad con los navegadores: Mozilla Firefox 2.0+, Intemet Explorer 6+, Safari 3+, Opera 10.6 y Google Chrome 8+.4
- Completa manipulación del árbol DOM.
- Es posible utilizarla con otras librerías como Prototype, gracias al modo que evita conflictos con las mismas.
- Manipulación de estilos CSS.
- Gestión de eventos de componentes HTML.
- Uso de efectos enriquecidos, animaciones y desarrollo de plugins personalizados (Ajax, funciones de autocompletar, elementos que aparecen y desaparecen, etc.)

JQuery. Ventajas frente a JavaScript

- Necesita menos código para determinadas tareas
- Tiene una gran variedad de efectos y utilidades sobre los elementos HTML
- Solventa problemas en la compatibilidad de los navegadores
- La manipulación del DOM se vuelve relativamente sencilla
- Permite una excelente integración con AJAX

JQuery. Empezando con jQuery

- Hay dos formas fundamentales de incluir la librería en nuestras páginas web:
    - **Descargar la última versión del framework e incluirlo en el documento HTML**
    - **Referenciando la librería a través de una versión online facilitada por alguna CDN**

JQuery. Empezando con jQuery

- **Descargar la última versión del framework e incluirlo en el HTML**
  - La página oficial desde la que se puede realizar la descarga es: https://jquery.com/download
  - La descarga consiste en un único archivo js que debemos almacenar en disco para la realización de las pruebas.
  - A continuación, basta con **crear un documento HTML e incluirlo de la siguiente manera:**

JQuery. Empezando con jQuery

- **Descargar la última versión del framework e incluirlo en el HTML**

```
<!DOCTYPE html>
<html>
    <head>
        <title>Ejemplo JQuery</title>
    </head>
    <body>
        <a href="http://jquery.com"/>jQuery</a>
        <script src="js/query.js"></script>
    </body>
</html>
```

  - **En el atributo src hay que indicar la ruta en la que se encuentra el fichero js de jQuery.**

JQuery. Empezando con jQuery

- **Referenciar la librería a través de versión online facilitada por una CDN**
  - Como ejemplo, la CDN de Google:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
```

  - O bien utilizar la CDN de jQuery: http://code.jquery.com/jquery-3.0.0.min.js

JQuery. Empezando con jQuery

- **Referenciar la librería a través de versión online facilitada por una CDN**
    - **Ventajas de este método:**
        - **Más velocidad**, pues los CDN tienen servidores repartidos por todo el mundo, es muy probable que la aplicación pueda obtener la información más rápidamente desde un sitio más cercano.
        - **Más sencilla**, simplemente tenemos que referenciar una URL

JQuery. Conceptos básicos

Ejecución de código una vez que la página ha sido cargada

- Cuando queremos realizar acciones con JavaScript que impliquen modificar ciertos aspectos de la página (crear elementos, quitarlos, etc.) es imprescindible aseguramos de que la página haya sido completamente cargada y se encuentre preparada para su manipulación.

JQuery. Conceptos básicos

Ejecución de código una vez que la página ha sido cargada

- jQuery proporciona una forma de realizar acciones justo cuando la página está lista para recibir instrucciones, pese a que puedan existir elementos que no hayan sido totalmente cargados mediante el bloque **$(document).ready()**

JQuery. Conceptos básicos

Ejecución de código una vez que la página ha sido cargada

```
$(document).ready(function(){
{
    $("a").click(function(event)){
        alert("Prueba del evento Click en un elemento a");
        event.preventDefault();
    });
});
```

Actividad 1

Basándonos en lo visto hasta ahora, crea una página HTML básica, pon correctamente las referencias y, utilizando JQuery, lanza una alerta con el texto:

"Hola Montecastelo 2020!"

JQuery. Conceptos básicos. Función $()

Función $()

- **La función básica de jQuery es la función dólar o función $()**, siendo la forma de interactuar con el HTML
- Es equivalente a la instrucción **document.getElementById().**

JQuery. Conceptos básicos. Función $()

Función $()

- **Recibe como parámetro una expresión o selector CSS**, o el nombre de una etiqueta HTML, **y como resultado devuelve todos los nodos que coincidan con dicha expresión**.
- A continuación, se expone cómo acceder a dichos elementos.

JQuery. Conceptos básicos. Función $()

- **Selección de elementos**
  - **En base al ID**
    - **$('#pfirst')**: devuelve el elemento con id pfirst, recordando que los IDs deben ser únicos por página
  - **En base a su class**
    - **$('.enable'):** devuelve los elementos cuya class sea 'enable'
    - **$('li.enable'):** devuelve todos los li cuya class sea 'enable'

JQuery. Conceptos básicos. Función $()

- **Selección de elementos**
  - **Por tipo de elemento**
    - **$('li’)**: devuelve una colección de ítems de una lista HTML
  - **Por su atributo**
    - **$('input[name=username]’)**: Devuelve aquellos elementos INPUT cuyo atributo name sea “username”

JQuery. Conceptos básicos. Función $()

- **Selección de elementos**
  - **Mediante selectores CSS**
    - **$('#center ul.clients li'):** devuelve los ítems li de una lista que estén centrados
  - **Selector asterisco**
    - **$('*'):** devuelve todos los elementos de la página

JQuery. Conceptos básicos. Función $()

- **Selección de elementos**
  - **Mediante pseudo-selectores**
    - **$('p[img)'):** selecciona todos los párrafos que contengan una imagen
    - **$('#login:input'):** selecciona todos los inputs del formulario login
    - **$("p:lt (3)"):** selecciona los 3 primeros párrafos del documento
    - **$("div:hidden"):** selecciona todos los divs ocultos

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Selectores de jerarquías**
    - **parent > child**: selecciona los elementos especificados como hijos del elemento padre.
    - **el1 el2**: selecciona todos los elementos 2 que descienden del elemento 1.
    - **el1 + el2**: obtiene los elementos 2 inmediatamente precedidos por el elemento 1.
    - **el1 ~ el2**: proporciona los elementos 2 precedidos por el elemento 1, aunque existan elementos intermedios.

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Selectores de jerarquías**
    - **:first-child**: obtiene aquellos elementos que son el primer hijo de otro elemento.
    - **:last-child**: aquellos elementos que son el último hijo de otro elemento.
    - **:nth-child(n)**: los elementos que son el enésimo hijo de su padre.
    - **:only-child**: los elementos que son el único hijo de su padre.

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Selectores posicionales**
    - **:first**: primer elemento.
    - **:last**: último elemento
    - **:not(selector)**: elementos que no coinciden con el selector.
    - **:even**: elementos impares, empezando por el cero.
    - **:odd**: elementos pares, empezando por el cero.
    - **:eq(n)**: selecciona el elemento que ocupe el puesto n en la lista
    - **:gt(n)**: elementos con un índice mayor que el n.
    - **:lt(n)**: elementos con un índice menor que el n.

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Filtrado de contenido**
    - **:contains(text)**: elementos que contienen el texto
    - **:empty**: elementos sin hijos (ni nodos de texto).
    - **:has(selector)**: elementos que contienen al menos un elemento que coincida con el selector
    - **:parent**: último elemento.
    - **:header**: selecciona los encabezados (hl, h2, etc).
    - **:animated**: elementos que forman parte de una animación.

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Selectores de atributo**
    - **:E[name]**: elementos que contienen el atributo con independencia de su valor.
    - **:E[name=value]**: elementos que contienen el atributo con el valor dado.
    - **:E[name^=value]:** elementos que contienen el atributo con el valor que empiece cadena especificada,
    - **:E[name$=value]:** elementos que contienen el atributo con el valor que termine por la cadena especificada.

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Selectores de atributo**
    - **:E[name*=value]:** elementos que contienen el atributo con el valor que contiene la subcadena especificada.
    - **:E[name |= value]:** elementos que contienen el atributo con el valor dado o que empieza por el valor dado seguido de un guion.
    - **:E[name != value]:** elementos que no contienen el atributo o que bien lo contienen, pero cuyo valor no coincide con el especificado.
    - **:E[name = value] [name2 = value2]:** elementos coincidentes con dichos filtros.

JQuery. Conceptos básicos. Función $()

- **Selectores avanzados**
  - **Filtrado por formulario**
    - **:hidden**: elementos que están ocultos
  - **Elementos de formulario**
    - **:input**: elementos input, textarea, select y button.
    - **:test**: inputs de tipo text
    - **:password**: inputs de tipo password

Actividad 2

Realizaremos un ejemplo con algunos de los filtros vistos hasta ahora

JQuery. Comprobar selección

- De otro tipo de lenguajes podríamos pensar que para realizar una comprobación de los elementos recuperados podríamos implementar con un condicional if:

```
if ($('div.central')) {... }
```

JQuery. Comprobar selección

- Sin embargo, de esta forma siempre se devuelve un objeto, incluso aunque la selección no contenga ninguno, por lo tanto, el resultado de la ejecución se evaluará siempre a True. Por esta razón, lo ideal será evaluar la condición a partir del método que devuelve la longitud del objeto:

```
if ($('div.central'). length) { ... }
```

JQuery. Guardar selecciones

- Al igual que en otros lenguajes podemos almacenar los elementos recuperados en variables a fin de poder manejar el resultado, y aplicar cualquier método jQuery deseado.

```
var $listItems = $(li);
```

Actividad 3

Basándonos en lo visto hasta ahora, crea una página HTML básica, pon correctamente las referencias y, utilizando JQuery, lanza una alerta con el texto:

"Hola Montecastelo 2020!"

JQuery. Filtros

- Podemos realizar selecciones sobre los elementos previamente seleccionados. Para ello, los filtros principales existentes son:
    - **.not()**: Para aquellos elementos que no cumplen con una condición.
    - **.filter()**: Selecciona los elementos que sí cumplen con una condición.
    - **.slice(init, fin)**: Selecciona los elementos contenidos entre el inicio y el fin, sin incluirlo este último.

JQuery. Filtros

- Podemos realizar selecciones sobre los elementos previamente seleccionados. Para ello, los filtros principales existentes son:

    - **.eq(index)**: De la selección total se queda con el que tenga el índice n (teniendo en cuenta comienza por el 0).
    - **.first()**: devuelve el primer elemento encontrado.
    - **.has()**: Selecciona un subconjunto de elementos que contengan el elemento especificado.

JQuery. Buscar elementos

- Los siguientes selectores permite interactuar con elementos:
  - **.next(), .nextAll(), .prev(), .prevAll()**: Permiten seleccionar los elementos hermanos posteriores y previos al elemento seleccionado.
  - **.parent(), .parents(), children(), siblings():** Permiten recuperar los padres, hijos y hermanos de un elemento.
  - **.contents():** Permite seleccionar todos los elementos hijos, incluido el texto de los mismos.

JQuery. Buscar elementos

- Los siguientes selectores permite interactuar con elementos:
  - **.add():** Permite añadir elementos al selector.
  - **.closest():** comprueba si el elemento actual coincide con la condición especificada para retornarlo. En caso contrario, seguirá realizando la búsqueda hacia arriba, padre a padre, hasta el que vea que coincide con dicho patrón.
  - **.find():** selecciona elementos descendientes del selector.

JQuery. Encadenamiento

- Es posible encadenar los diferentes métodos expuestos anteriormente:

```
$('#central').find('p').eq(3).html('texto del cuarto párrafo');
```

  - El método **end()** permite reestablecer la selección a todos los elementos seleccionados antes de haber realizado ciertas modificaciones sobre la misma.

  - El método **andBack()** permitiría seleccionar también el elemento del DOM anterior al actual.

JQuery. Obtener y establecer contenido

- Existen tres métodos muy utilizados para manipular el DOM del documento mediante jQuery:
  - **text():** retorna o establece el texto contenido en los elementos.
  - **html():** retorna o establece el texto contenido en los elementos incluso con marcado
  - **val():** retorna o establece el valor de los campos del formulario.

JQuery. Obtener y establecer contenido

- Su uso es muy sencillo:
  - Para establecer:

```
$('h1').html('hello world');
```

  - Para obtener el valor:

```
$('h1').html();
```

  - Si lo que queremos es obtener o establecer el valor de atributos de un elemento:

```
attr();
```

JQuery. Obtener y establecer contenido

- Un ejemplo completo:

```
$("button").click(function ( )
{
    alert($("#btnl").attr("href")); //Retorna el valor del href
    $("#btnl").attr("href", "www.google.es"); //Establece el valor de href
});
```

JQuery. Añadir contenido

- Se puede utilizar alguno de los siguientes métodos:
  - **append**(): inserta contenido al final de los elementos seleccionados
  - **prepend**(): inserta contenido previamente a los elementos seleccionados
  - **after**(): permite insertar contenido después de los elementos seleccionados
  - **before**(): permite insertar contenido antes de los elementos seleccionados

JQuery. Eliminar contenido

- Para eliminar elementos o contenido se pueden utilizar principalmente dos métodos:
  - **remove**(): elimina elementos seleccionados y sus hijos.
  - **empty**(): elimina los hijos de los elementos seleccionados

JQuery. Manipular CSS

- Entre otros métodos, jQuery provee los siguientes métodos para manipular la CSS:
  - **addClass**(): añade una o más clases a los elementos seleccionados
  - **removeClass**( ) : elimina una o más clases a los elementos seleccionados
  - **toggleClass**( ): cambia entre la adición / eliminación de las clases de los elementos seleccionados
  - **css**(): establece o devuelve el atributo de estilo

JQuery. Atributos

- Una vez seleccionado un elemento es posible acceder a sus atributos para actuar sobre él.
- Los atributos que puedes utilizar son:
    - .**addClass**() : añade una clase CSS al elemento.
    - .**attr**() : obtiene o establece un valor a un atributo determinado.
    - .**hasClass**(): determina si un elemento tiene asignado una clase CSS determinada.

JQuery. Atributos

- Los atributos que puedes utilizar son:
    - .**html**(): obtiene o establece el contenido HTML dentro de un elemento.
    - .**removeAttr**() : borra un atributo de un elemento
    - .**removeClass**(): borra una clase CSS de un elemento.
    - .**removeProp**(): borra una propiedad de un elemento.

JQuery. Atributos

- Los atributos que puedes utilizar son:
    - .**toggleClass**(): dependiendo de si la clase CSS ya estaba presente o no, añade o borra una o más clases CSS sobre un elemento.
    - .**val**(): obtiene o establece el valor de un elemento.
- Se puede consultar toda esta información en el siguiente enlace: http://docs.jquery.com/Attributes

JQuery. Atributos. Recorrido de colecciones

- jQuery nos ofrece también una manera mucho más natural de recorrer colecciones de elementos con una sintaxis más simple que el clásico bucle for.
- La sintaxis sería la siguiente:

JQuery. Atributos. Recorrido de colecciones

```
$.each([50, 90], function (index, value) {
    alert(index + ': ' + value);
});
```

- Siendo [50, 90] un array de dos elementos enteros, index el índice del elemento del array en el que estamos en cada iteración, y value el elemento en el que estamos (en la primera iteración valdrá 50 y en la segunda 90).
- No es necesario inicializar ningún contador (index se actualiza solo).

JQuery. Atributos. Manipulación del DOM

- jQuery también proporciona una serie de métodos para manipular el DOM del documento de una forma muy sencilla y directa.
- Algunos de esos métodos son:
    - **.after():** Inserta contenido después del elemento seleccionado.
    - **.append():** Inserta contenido al final del elemento seleccionado.
    - **.before():** Lo contrario a after().

JQuery. Atributos. Manipulación del DOM

- Algunos de esos métodos son:
    - **.prepend():** Inserta contenido al principio del elemento seleccionado.
    - **.css():** Obtiene o establece el valor de una propiedad CSS en el elemento seleccionado.
    - **.empty():** Borra todos los nodos hijos del elemento seleccionado.
    - **.height():** Obtiene la altura del elemento seleccionado.

JQuery. Atributos. Manipulación del DOM

- Algunos de esos métodos son:
    - **.remove():** borra el elemento seleccionado del DOM.
    - **.replaceAll():** Reemplaza los elementos seleccionados con un elemento dado.
- Se puede encontrar la lista completa en el siguiente enlace:

http://docs.jquery.com/Manipulation

JQuery. Eventos

- Hasta ahora hemos visto cómo **manejar eventos en JavaScript** añadiendo un elemento HTML, como por ejemplo el onclick apuntando a una función Javascript que será su manejador. Por ejemplo:

```
<a id="link" href="#" onclick="functionName('value')"> Click aquí </a>
```

JQuery. Eventos

- **Para manejar eventos con jQuery** tan solo debemos seleccionar el o los elementos a los que asignarle un manejador, e indicar qué tipo de evento y qué función será el manejador, pudiendo indicarse todo en la misma instrucción. Por ejemplo, para el enlace anterior:

```
$("#link").click(function () {
    alert("Hola");
});
```

JQuery. Eventos

- Los eventos más habituales son:
    - **.bind():** Sirve para asignar un evento a un manejador de eventos.
    - **.blur():** Se lanza cuando un elemento pierde el foco de selección
    - **.change():** Se lanza cuando cambia el valor de un elemento. Puede ser lanzado por los controles:
        - Input
        - Textarea
        - Select
    - **.click()**: Se dispara cuando se hace click en el evento seleccionado

JQuery. Eventos

- Los eventos más habituales son:
  - **.delegate():** Similar a bind(), pero con este podemos asignar eventos a elementos que aún no se encuentran en la página HTML.
  - **.event.preventDefault():** Al llamar a este evento estamos cancelando el comportamiento por defecto de una determinada acción.
  - **.focus():** Este evento se dispara cuando un elemento recibe el foco de selección.
  - **.hover():** Se dispara cuando el puntero del ratón pasa por encima del elemento indicado.

JQuery. Eventos

- Los eventos más habituales son:
    - **.on():** Vincula manejadores de eventos al elemento seleccionado. Se puede revertir el efecto de on() con el off()
    - **.ready():** Se ejecutará una vez que la página haya terminado de cargarse.
    - **.submit():** Se dispara cuando se intenta enviar un formulario. Por tanto, este evento solo puede ser asociado a elementos form.
- Información de todos los eventos: http://api.jquery.com/category/Events

JQuery. Efectos

- Se pueden crear efectos sobre los elementos de las páginas HTML.
- Los más usados son:
    - **show**(): Muestra uno o varios elementos seleccionados si en ese momento están ocultos.
    - **hide**(): Oculta un elemento que en ese momento se está mostrando.
    - **toggle**(): Es una combinación de los dos anteriores. toggle() oculta un elemento visible, y lo muestra si no está visible

JQuery. Efectos

- Los más usados son:
    - **slideDown**(): Hace aparecer un elemento en la página mediante una cortina descendente.
    - **slideUp**(): Hace aparecer un elemento en la página mediante una cortina ascendente.
    - **fadeIn**(): Hace aparecer un elemento en la página poco a poco ajustando su opacidad.
    - **fadeOut**(): Hace desaparecer un elemento en la página poco a poco ajustando su opacidad.
- Lista completa de eventos: http://docs.jquery.com/Effects

Actividad 3

Basándonos en lo visto hasta ahora, crea una página HTML básica, pon correctamente las referencias y, utilizando JQuery, lanza una alerta con el texto:

"Hola Montecastelo 2020!"

JQuery Validation

- JQuery también nos proporciona herramientas de validación en formularios que aceleran, facilitan y hacen mucho más limpio este tipo de código
- **Como se comprueba la validez de los datos sin necesidad de que estos viajen al servidor se gana muchísimo en eficiencia**

JQuery Validation

- Para su uso debemos:
    - Descargar el plugin de validación **jquery.validate.min.js** (como en otras ocasiones, está disponible en la página oficial o bien referenciando la CDN)
    - Incluirlo en la carpeta /js de nuestro directorio web.
    - Utilizar el método **validate** sobre el formulario que queremos validar

JQuery Validation

- El método **validate** funciona con:
    - **Rules:** mediante métodos se puede establecer las reglas de validación que empleará cada campo que será referenciado por id: required, lettersonly, digits, minlength, maxlength, email, remote, equalTrue, iban, etc.

```
var validator = $("#formularioRegistro").validate((
rules: {
nombre: "required",
        apellido: "required",
        login: {
        required: true,
                minlength: 8
        },
```

JQuery Validation

- El método **validate** funciona con:
    - **Messages**: Permite personalizar los mensajes de error que queremos que proporcione al usuario en caso de que el campo no cumpla la regla de validación.

```
messages: {
    nombre: "Introduzca su nombre",
    apellido: "Introduzca su apellido",
    login: {
        required: "Introduzca un usuario",
        minlength: jQuery.format("Introduzca al menos (0) caracteres")
    },
```

JQuery Validation

- El método **validate** funciona con:
    - Propiedades **errorPlacement**, **errorContainer**, **errorLabelContainer**, **wrapper**, **errorElement** se especifica dónde y cómo formatear los mensajes de error. Son opcionales.

```
errorPlacement: function(error, element){
if (element.is(":radio"))
        error.appendTo(element.parent().next().next());
        else if (element.is(":checkbox"))
        error.appendTo(element.next());
        else
        error.appendTo(element.parent().next());
        },
```

JQuery Validation

- El método **validate** funciona con:
    - **submitHandler**: Permite especificar cualquier función o instrucción a realizar una vez que se envía el formulario.

```
submitHandler: function() {
    alert("submitted!");
}
```

Actividad 3

Un ejemplo de Jquery Validation o así

Componentes Bootstrap con JavaScript

- En Bootstrap existen componentes dinámicos desarrollados sobre JQuery listos para ser usados.
- Los componentes hacen uso de los atributos data-* que HTML5 nos permite definir de forma libre en cualquier etiqueta de la página.
- Veremos alguno de los componentes a continuación:

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Alert

  - Permiten mostrar pequeños mensajes cuyo nivel de gravedad puede ser modulado mediante colores

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="alert alert-success" role="alert">
    Alerta de mensaje de éxito
</div>
<div class="alert alert-danger" role="alert">
    Alerta de mensaje de peligro
</div>
<div class="alert alert-warning" role="alert">
    Alerta de mensaje de warning
</div>
<div class="alert alert-info" role="alert">
    Alerta de mensaje de información
</div>
```

- **Resultado:**

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Tooltip

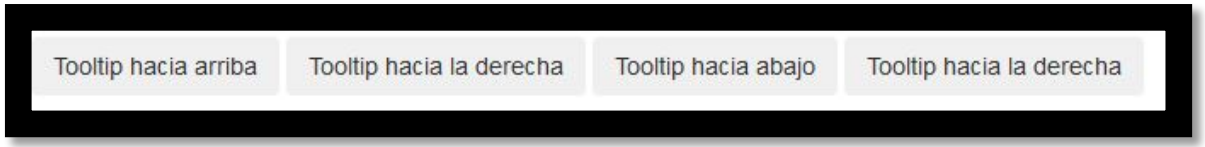
  - Es un componente que debe ser inicializado de forma explícita.

Componentes Bootstrap con JavaScript

- **HTML**:

```html
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="top" title="Tooltip hacia arriba">
    Tooltip hacia arriba
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="right" title="Tooltip hacia la derecha">
    Tooltip hacia la derecha
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="bottom" title="Tooltip hacia abajo">
    Tooltip hacia abajo
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="left" title="Tooltip hacia la izquierda">
    Tooltip hacia la derecha
</button>
```

- **Resultado:**



Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Popover

  - Son similares a los Tooltips pero pueden incluir cualquier contenido HTML.

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Pestañas o Tabs

  - Es un componente que debe ser inicializado de forma explícita.

Componentes Bootstrap con JavaScript

- **HTML**:

```
<ul class="nav nav-tabs" id="myTab" role="tablist">
    <li class="nav-item">
        <a class="nav-link active" id="home-tab" data-toggle="tab" href="#home" role="tab" aria-controls="home"
           aria-selected="true">Home</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" id="profile-tab" data-toggle="tab" href="#profile" role="tab" aria-controls="profile"
           aria-selected="false">Perfil</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" id="contact-tab" data-toggle="tab" href="#contact" role="tab" aria-controls="contact"
           aria-selected="false">Contacto</a>
    </li>
</ul>
```

- **Resultado:**

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Ventana modal

  - Son las ventanas pop-up que ofrece Bootstrap

- **Javascript**

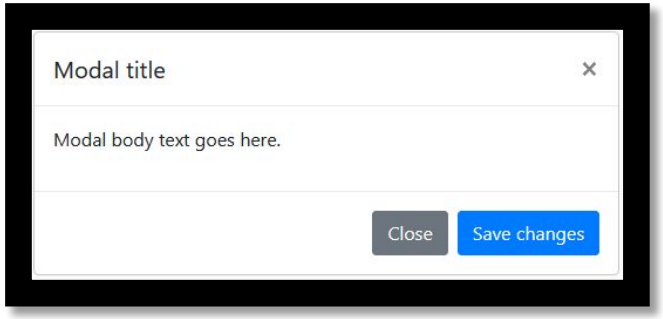
```
$('#miVentanaModal').on('shown.bs.modal', function () {
    $('#miBoton').trigger('focus')
});
```

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="modal" tabindex="-1" role="dialog">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title">Modal title</h5>
                <button type="button" class="close" data-dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                <p>Modal body text goes here.</p>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-primary">Save changes</button>
                <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>
```

- **Resultado:**



Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Carrusel

  - Conjunto habitual que se muestra en páginas de inicio: conjunto de imágenes con un mensaje breve que permiten destacar determinada información de forma espectacular.

Componentes Bootstrap con JavaScript

- **HTML**:

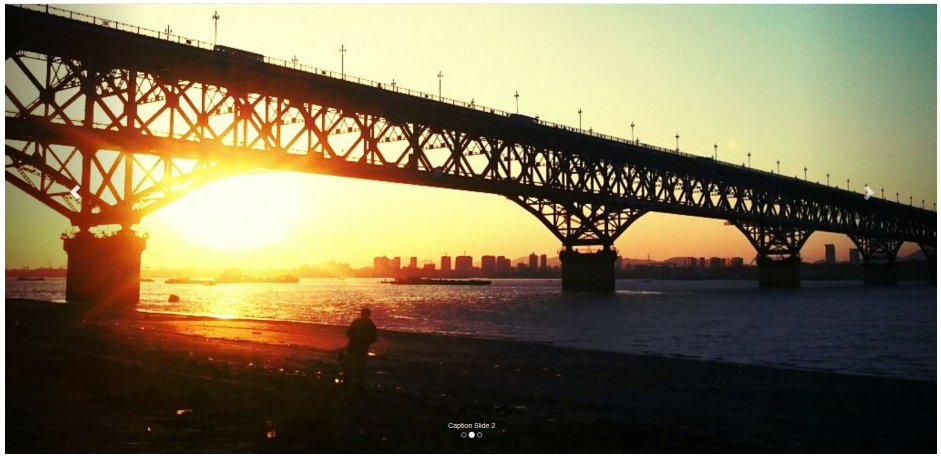
```
<div id="carousel-example-generic" class="carousel slide" data-ride="carousel">
    <!-- Indicators -->
    <ol class="carousel-indicators">
        <li data-target="#carousel-example-generic" data-slide-to="0" class="active"></li>
        <li data-target="#carousel-example-generic" data-slide-to="1"></li>
        <li data-target="#carousel-example-generic" data-slide-to="2"></li>
    </ol>

    <!-- Wrapper for slides -->
    <div class="carousel-inner" role="listbox">
        <div class="item active">
            <img src="img/image.jpg" alt="Slide 1">
            <div class="carousel-caption">
                Caption Slide 1
            </div>
        </div>
        <div class="item">
            <img src="img/image.jpg" alt="Slide 2">
            <div class="carousel-caption">
                Caption Slide 2
            </div>
        </div>
        <div class="item">
            <img src="img/image.jpg" alt="Slide 3">
            <div class="carousel-caption">
                Caption Slide 3
            </div>
        </div>
    </div>

    <!-- Controls -->
    <a class="left carousel-control" href="#carousel-example-generic" role="button" data-slide="prev">
        <span class="glyphicon glyphicon-chevron-left" aria-hidden="true"></span>
        <span class="sr-only">Previous</span>
    </a>
    <a class="right carousel-control" href="#carousel-example-generic" role="button" data-slide="next">
        <span class="glyphicon glyphicon-chevron-right" aria-hidden="true"></span>
        <span class="sr-only">Next</span>
    </a>
</div>
```

Componentes Bootstrap con JavaScript

- **Resultado:**



Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Acordeón

  - Es una sección que aparece y desaparece al pulsar un botón.

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="accordion">
    <div class="accordion-section">
        <a class="accordion-section-title" href="#accordion-1">Sección #1</a>
        <div id="accordion-1" class="accordion-section-content">
            <p>
                Lorem ipsum dolor sit amet, consectetur adipiscing elit.
            </p>
        </div>
    </div>
</div>
```

Componentes Bootstrap con JavaScript

- **CSS**:

```
.accordion
{
    overflow:hidden;
    border-radius:4px;
    background:#f7f7f7;
}
.accordion-section-title
{
    width:100%;
    padding:15px;
}
.accordion-section-title
{
    width: 100%;
    padding: 15px;
    display: inline-block;
    background-color: #333;
    border-bottom: 1px solid #1a1a1a;
    font-size: 1.2em;
    color: #fff;
    transition: all linear 0.5s;
    text-decoration:none;
}
.accordion-section-title.active
{
    background-color:#4c4c4c;
    text-decoration:none;
}
.accordion-section-title:hover
{
    background-color:grey;
    text-decoration:none;
}
.accordion-section:last-child .accordion-section-title
{
    border-bottom:none;
}
.accordion-section-content
{
    padding:15px;
    display:none;
}
```

Componentes Bootstrap con JavaScript

- **Javascript**:

```
$(document).ready(function () {
    $('.accordion-section-title').click(function (e) {
        var currentAttrvalue = $(this).attr('href');
        if ($(e.target).is('.active')) {
            $(this).removeClass('active');
            $('.accordion-section-content:visible').slideUp(300);
        } else {
            $('.accordion-section-title').removeClass('active').filter(this).addClass('active');
            $('.accordion-section-content').slideUp(300).filter(currentAttrvalue).slideDown(300);
        }
    });
});
```

Componentes Bootstrap con JavaScript

- **Resultado**:

- **Resultado al clicar:**

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Menú Dropdown

  - Los menús desplegables son muy útiles para ofrecer opciones, menús laterales, etc.

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="container">
    <h2>Ejemplo Dropdown</h2>
    <p>Clicando sobre el botón se abrirá el menú</p>
    <div class="dropdown">
        <button class="btn btn-primary dropdown-toggle" type="button" data-toggle="dropdown">Lenguajes Web
            <span class="caret"></span></button>
        <ul class="dropdown-menu">
            <li><a href="#">HTML</a></li>
            <li><a href="#">CSS</a></li>
            <li><a href="#">JavaScript</a></li>
        </ul>
    </div>
</div>
```

- **Resultado:**

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Barras de progreso

  - Ofrece una información importante al usuario, ya que puede hacerse una idea de que una tarea está en curso y además puede conocer su progreso.

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="container">
    <h2>Porcentaje completado</h2>
    <div class="progress">
        <div class="progress-bar" role="progressbar" aria-valuenow="70" aria-valuemin="0" aria-valuemax="100" style="width:90%">
            <span class="sr-only">90% Complete</span>
        </div>
    </div>
</div>
```

- **Resultado:**

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Formularios

  - Existen distintas formas de maquetar el formulario.
  - Veremos ahora el **formulario básico**

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="container">
    <h2>Formulario básico</h2>
    <form action="/action_page.php">
        <div class="form-group">
            <label for="email">Email:</label>
            <input type="email" class="form-control" id="email" placeholder="Introduce email" name="email">
        </div>
        <div class="form-group">
            <label for="pwd">Password:</label>
            <input type="password" class="form-control" id="pwd" placeholder="Introduce password" name="pwd">
        </div>
        <button type="submit" class="btn btn-default">Enviar</button>
    </form>
</div>
```

- **Resultado:**

Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Formularios

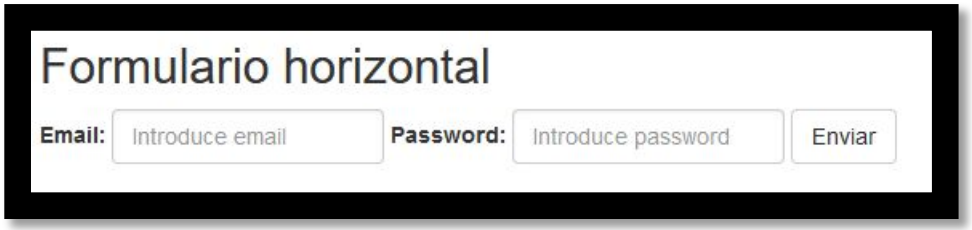
  - Existen distintas formas de maquetar el formulario.
  - Veremos ahora el **formulario inline**

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="container">
    <h2>Formulario inline</h2>
    <form class="form-inline" action="/action_page.php">
        <div class="form-group">
            <label for="email">Email:</label>
            <input type="email" class="form-control" id="email" placeholder="Introduce email" name="email">
        </div>
        <div class="form-group">
            <label for="pwd">Password:</label>
            <input type="password" class="form-control" id="pwd" placeholder="Introduce password" name="pwd">
        </div>
        <button type="submit" class="btn btn-default">Enviar</button>
    </form>
</div>
```

- **Resultado:**



Componentes Bootstrap con JavaScript

- Algunos de los componentes son:

Formularios

  - Existen distintas formas de maquetar el formulario.
  - Veremos ahora el **formulario horizontal**

Componentes Bootstrap con JavaScript

- **HTML**:

```
<div class="container">
    <h2>Formulario horizontal</h2>
    <form class="form-horizontal" action="/action_page.php">
        <div class="form-group">
            <label class="control-label col-sm-2" for="email">Email:</label>
            <div class="col-sm-10">
                <input type="email" class="form-control" id="email" placeholder="Introduce email" name="email">
            </div>
        </div>
        <div class="form-group">
            <label class="control-label col-sm-2" for="pwd">Password:</label>
            <div class="col-sm-10">
                <input type="password" class="form-control" id="pwd" placeholder="Introduce password" name="pwd">
            </div>
        </div>
        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <button type="submit" class="btn btn-default">Enviar</button>
            </div>
        </div>
    </form>
</div>
```

- **Resultado:**

Actividad 1

Basándonos en lo visto hasta ahora, crea una página HTML básica, pon correctamente las referencias y, utilizando JQuery, lanza una alerta con el texto:

"Hola Montecastelo 2020!"

Tema 6: Integración de contenido interactivo

Ciclo Superior DAW

Asignatura: Diseño de interfaces

Curso 20/21