



JQUERY

© 2017, ACTIBYTI PROJECT SLU, Barcelona
Autor: Ricardo Ahumada



MINISTERIO
DE ENERGÍA, TURISMO
Y AGENDA DIGITAL

red.es



ESTRATEGIA DE
EMPRENDIMIENTO Y
EMPLEO JUVENIL
garantía juvenil



UNIÓN EUROPEA

Fondo Social Europeo
“El FSE invierte en tu futuro”

ÍNDICE DE CONTENIDOS

1. Introducción
2. Selección
3. Manipulación DOM
4. Eventos
5. Efectos
6. Validación de formularios

1

INTRODUCCIÓN

Jquery \$()

- Potente librería JS creada por John Resig.
- Proyecto Open-source, liberado en 2006
- Provee características nuevas a Javascript
 - Sintaxis más simple
 - Curva de aprendizaje más corta
 - Robusta compatibilidad de plataforma cruzada
 - Compatibilidad con navegadores mobile y de escritorio
(<https://jquery.com/browser-support/>)



Utilidades y características

- Acceder al Documento HTML (DOM: Document Object Model)
- Modificar la apariencia de la página
- Modificar el contenido de la página
- Manejar eventos en la página
- Crear efectos visuales
- Manipular estilos CSS
- Procesar órdenes Ajax
- Simplifica tareas comunes: Manipulación de arrays, iteración, operación con objetos
- Manipulación de JSON
- Programación bajo “Paradigma no invasivo”
- Sistema de plugins que extienden sus funcionalidad

Como funciona

- `$()` o `jQuery()` crean un objeto
- `$('a')` es un objeto que contiene todos los elementos `a`
- jQuery crea un punto de acción entre el tiempo que se carga el DOM y las imágenes `$('document').ready(f())`
- Este punto es más eficaz que `<body onload="f()">`

Incluir Jquery en la página

- La librería se puede descargar de: <http://jquery.com/download/>
- Una vez descomprimido, se debe incluir la referencia en el html

```
<html>
  <head>
    <title>The jQuery Example</title>
  </head>

  <body>
  ...
  </body>
  <script type = "text/javascript" src = "/jquery/jquery-2.1.3.min.js"></script>
</html>
```

- También se puede incluir la referencia en el html de sitio google

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

Programación no invasiva

- JQuery permite separar eficientemente el HTML del código Javascript
- Esto se logra a través del método ready sobre el DOM

```
<html>

  <head>
    <title>The jQuery Example</title>
  </head>

  <body>
    <h1>Hello</h1>
  </body>

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script type = "text/javascript">
    $(document).ready(function(){
      alert("Hello, World!");
    });
  </script>
</html>
```



Pongámoslo en práctica

- Crea un nuevo proyecto web
- Añade jquery al html
- Haz que cuando el documento se haya cargado, aparezca el mensaje “documento cargado”

2

SELECCIÓN

Selectores

- La forma básica de interactuar con la página es mediante la función `$(())`, alias de `jQuery()`
- Esta función recibe como parámetro una expresión CSS o el nombre de una etiqueta

```
<div id="#more">more <a href="#" class="boton">Enviar</a></div>
```

```
$(document).ready(function(){
    $('#more');
    $('.boton');
});
```

- Esta selección generará un objeto (array) que expondrá una API de métodos para actuar sobre los elementos seleccionados
- <http://api.jquery.com/>

Selectores CSS

- Nombre de elemento `$(‘p’)`
- Id `$(‘#id_elemento’)`
- Clase `$(‘.clase’)`
- Elemento con clase `$(‘p.clase’)`
- Elemento descendiente de un elemento `$(‘p a’)`
- Elemento hijo de un elemento `$(‘p > a’)`

Selectores Xpath y pseudoclases

- \$('a[@href]') > Link con atributo href
- \$('div[ol]') > Div con un dentro
- \$('a[@rel="nofollow"]') > Link con valor de atributo específico
- P:first > El primer P
 - Se puede combinar con otros elementos
 - \$(div#body p:first)
- Li:last > El último li
- A:nth(3) o A:eq(3) > El cuarto link
- P:even > Toma los elementos impares
- P:odd > Toma los elementos pares
- A:gt(3), a:lt(3) > Todos los elementos mayores que 3 o menores.
- A:contains('click') > Todos los a que contienen la palabra click
- P:hidden > Elementos p ocultos



Pongámoslo en práctica

- Crea una página con dos secciones
 - En la primera sección generar un artículo que contenga título, imágenes, listas. Indica para algunos id, para otros class. En el caso de las listas déjalos sin id o clase.
 - En la segunda sección añade un formulario de alta de usuario.
- Usando selectores de jQuery
 - Seleccionar elementos concretos de la página y examinalos en consola
 - Modifica el cuerpo de los elementos seleccionados mediante el método **.html(valor)**
 - Por ejemplo: \$('#I1').html('Elemento modificado 2');

3

MANIPULACIÓN DOM

Atributos

- El método **attr()** se puede utilizar para buscar el valor de un atributo (del primer elemento del conjunto combinado) o establecer valores de atributo en todos los elementos coincidentes.

```
var title = $("div").attr("title");
```

```
$("#myimg").attr("src", "/jquery/images/jquery.jpg");
```

Métodos de atributos

➤ A continuación se muestran algunos métodos útiles

Método	Descripción
attr(propiedades)	Establezca un objeto clave/valor ({property1:value1, property2:value2}) como propiedades para todos los elementos coincidentes.
attr(clave, fn)	Establezca una propiedad única en un valor calculado, en todos los elementos coincidentes.
removeAttr(nombre)	Elimine un atributo de cada uno de los elementos coincidentes.
hasClass(clase)	Devuelve true si la clase especificada está presente en al menos uno de los conjuntos de elementos coincidentes.
removeClass(clase)	Elimina todas o las clases especificadas del conjunto de elementos coincidentes.
toggleClass(clase)	Agrega la clase especificada si no está presente, elimina la clase especificada si está presente.
html()	Obtiene el contenido html (innerHTML) del primer elemento.
html(val)	Establezce el contenido html de cada elemento coincidente.
text()	Obtiene el contenido de texto combinado de todos los elementos coincidentes.
text(val)	Establece el contenido de texto de todos los elementos coincidentes.
val()	Obtiene el valor de entrada del primer elemento coincidente.
val(valor)	Establece el atributo de valor de cada elemento si se llama a <input>. Si se llama en <select> indicando la opción <option>, esta es seleccionada. Si se llama en check box o radio, se marcan todas las casilla de verificación coincidentes o la casilla de radio.

Estilos

- Se logra a través del método **addClass(classes)**, que permite establecer estilos definidos en un css.
- Se pueden aplicar múltiples estilos indicando las clases separadas por espacio.

```
$("p").addClass("selected");
```

```
$("#myid").addClass("highlight");
```

Filtrado de elementos

- jQuery proporciona varios métodos potentes para filtrar elementos del DOM.

Método	Descripción
eq(index)	Reduce el conjunto de elementos adaptados a un solo elemento.
filter(selector)	Elimina todos los elementos del conjunto de elementos coincidentes que no coinciden con el selector especificado.
filter(fn)	Elimina todos los elementos del conjunto de elementos coincidentes que no coinciden con la función especificada.
is(selector)	Comprueba la selección actual contra una expresión y devuelve true si al menos un elemento de la selección se ajusta al selector dado.
map(callback)	Traduce un conjunto de elementos en el objeto jQuery en otro conjunto de valores en una matriz jQuery (que puede, o no, contener elementos).
not(selector)	Elimina los elementos que coinciden con el selector especificado del conjunto de elementos coincidentes.
slice(start, [end])	Selecciona un subconjunto de los elementos.

Ejemplos

```
<ul>
  <li>list item 1</li>
  <li>list item 2</li>
  <li>list item 3</li>
  <li>list item 4</li>
  <li>list item 5</li>
  <li>list item 6</li>
</ul>
```

```
$( "li" ).eq(2).addClass("selected");
```

```
$( "li" ).filter(".middle").addClass("selected");
```

```
<p>This is 1st paragraph and <span>THIS IS RED</span></p>
<p>This is 2nd paragraph and <span>THIS IS ALSO RED</span></p>
```

```
$( "p" ).find("span").addClass("selected");
```

DOM Traversing

- Asimismo jQuery proporciona varios métodos potentes para recorrer el DOM.

Método	Descripción
add(selector)	Añade más elementos, emparejados por el selector dado, al conjunto de elementos coincidentes.
andSelf()	Añade la selección anterior a la selección actual.
children([selector])	Obtiene un conjunto de elementos que contengan todos los hijos inmediatos únicos de cada uno de los conjuntos de elementos coincidentes.
closest(selector)	Obtiene un conjunto de elementos que contienen el elemento principal más cercano que coincide con el selector especificado, incluido el elemento inicial.
contents()	Encuentra todos los nodos secundarios dentro de los elementos coincidentes(incluidos los nodos de texto) o el documento de contenido, si el elemento es un iframe.
end()	Revertir la operación 'destructiva' más reciente, cambiando el conjunto de elementos coincidentes a su estado anterior.
find(selector)	Busca elementos descendientes que coincidan con los selectores especificados.
next([selector])	Obtiene un conjunto de elementos que contengan los hermanos únicos de cada uno de los conjuntos de elementos.

DOM Traversing (II)

Método	Descripción
nextAll([selector])	Encuentra todos los elementos hermanos después del elemento actual.
offsetParent()	Devuelve una colección jQuery con el padre posicionado del primer elemento coincidente.
parent([selector])	Obtiene el parent directo de un elemento. Si se invoca un conjunto de elementos, parent devuelve un conjunto de sus elementos parent directos únicos.
parents([selector])	Obtiene un conjunto de elementos que contienen los antepasados únicos del conjunto de elementos coincidentes(excepto el elemento raíz).
prev([selector])	Obtiene un conjunto de elementos que contengan los hermanos anteriores únicos de cada uno de los conjuntos de elementos coincidentes.
prevAll([selector])	Encuentra todos los elementos hermanos delante del elemento actual.
siblings([selector])	Obtiene un conjunto de elementos que contengan todos los hermanos únicos de cada uno de los conjuntos de elementos coincidentes.

Ejemplos

```
<ul>
<li class = "above">list item 0</li>
<li class = "top">list item 1</li>
<li class = "top">list item 2</li>
<li class = "middle">list item 3</li>
<li class = "middle">list item 4</li>
<li class = "bottom">list item 5</li>
<li class = "bottom">list item 6</li>
<li class = "below">list item 7</li>
</ul>
```

```
$(".top").add(".middle").addClass("selected");
```

```
($("#ul").children(".selected").addClass("blue");
```

```
($("#li").siblings('.selected').addClass("hilight");
```

```
($("#li").parent().addClass('red');
```

Métodos CSS

- › jQuery admite casi todos los selectores incluidos en las especificaciones 1 a 3 de CSS

Método	Descripción
css(nombre)	Devuelve una propiedad de estilo en el primer elemento coincidente.
css(nombre, valor)	Establece una propiedad de estilo único en un valor de todos los elementos coincidentes.
css(properties)	Establece un objeto de clave/valor ({key1:val1, key2:val2....keyN:valN}) como propiedades de estilo para todos los elementos coincidentes.
height(val)	Establece la altura CSS de cada elemento coincidente.
height()	Obtiene la actual calculada, píxel, la altura del primer elemento emparejado.
innerHeight()	Obtiene la altura interna (excluye el borde e incluye el relleno) para el primer elemento coincidente.
innerWidth()	Obtiene el ancho interno (excluye el borde e incluye el relleno) para el primer elemento coincidente.
offset()	Obtiene el desplazamiento actual del primer elemento coincidente, en píxeles, en relación con el documento.
offsetParent()	Devuelve una colección jQuery con el padre posicionado del primer elemento coincidente.

Métodos CSS (II)

Método	Descripción
outerHeight([margen])	Obtiene la altura externa (incluye el borde y el relleno de forma predeterminada) para el primer elemento coincidente.
outerWidth([margen])	Obtenga el ancho exterior (incluye el borde y el relleno de forma predeterminada) para el primer elemento coincidente.
position()	Obtiene la posición superior e izquierda de un elemento en relación con su padre de desplazamiento.
scrollLeft(val)	Cuando se pasa un valor, el desplazamiento a la izquierda se ajusta a ese valor en todos los elementos coincidentes.
scrollLeft()	Obtiene el desplazamiento a la izquierda del primer elemento coincidente.
scrollTop(val)	Cuando se pasa un valor, el desplazamiento superior de desplazamiento se establece en ese valor en todos los elementos coincidentes.
scrollTop()	Obtiene el desplazamiento de la parte superior de desplazamiento del primer elemento coincidente.
width(val)	Establece el ancho CSS de cada elemento coincidente.
width()	Obtiene el ancho actual calculado, píxel, del primer elemento emparejado.

Ejemplos

```
<ul>
  <li>list item 1</li>
  <li>list item 2</li>
  <li>list item 3</li>
  <li>list item 4</li>
  <li>list item 5</li>
  <li>list item 6</li>
</ul>
```

```
$( "li" ).eq(2).css("color", "red");
```

```
$( "li" ).eq(2).css({ "color": "red",
                      "background-color": "green" });
```

```
$( "div:first" ).width(100);
$( "div:first" ).css("background-color", "blue");
```

Manipulación del DOM

- jQuery provee métodos para manipular el DOM de manera eficiente.

Método	Descripción
after(contenido)	Inserta contenido después de cada uno de los elementos coincidentes.
append(contenido)	Añade contenido al interior de cada elemento coincidente.
appendTo(selector)	Añade todos los elementos emparejados a otro, especificado, conjunto de elementos.
before(contenido)	Inserta el contenido antes de cada uno de los elementos coincidentes.
clone(bool)	Clona elementos DOM, y todos sus manejadores de eventos; además selecciona los clones.
clone()	Clona elementos DOM y seleccionó los clones.
empty()	Elimina todos los nodos secundarios del conjunto de elementos coincidentes.
html(val)	Establece el contenido html de cada elemento coincidente.
html()	Obtiene el contenido html (innerHTML) del primer elemento emparejado.
insertAfter(selector)	Inserta todos los elementos coincidentes después de otro, especificado, conjunto de elementos.

Manipulación del DOM (II)

Método	Descripción
insertBefore(selector)	Inserta todos los elementos coincidentes antes de otro, especificado, conjunto de elementos.
prepend(contenido)	Prepone contenido al interior de cada elemento coincidente.
prependTo(selector)	Prepone todos los elementos emparejados a otro, especificado, conjunto de elementos.
remove(expr)	Elimina todos los elementos coincidentes del DOM.
replaceAll(selector)	Reemplaza los elementos que coinciden con el selector especificado con los elementos coincidentes.
replaceWith(contenido)	Reemplaza todos los elementos coincidentes con los elementos HTML o DOM especificados.
text(val)	Establece el contenido de texto de todos los elementos coincidentes.
text()	Obtiene el contenido de texto combinado de todos los elementos coincidentes.
wrap(elem)	Envuelve cada elemento coincidente con el elemento especificado.
wrap(html)	Envuelve cada elemento coincidente con el contenido HTML especificado.
wrapAll(elem)	Envuelve todos los elementos del conjunto emparejado en un solo elemento de envoltura.

Manipulación del DOM (III)

Método	Descripción
<code>wrapAll(html)</code>	Envuelve todos los elementos del conjunto emparejado en un solo elemento de envoltura.
<code>wrapInner(elem)</code>	Envuelve el contenido secundario interno de cada elemento coincidente (incluidos los nodos de texto) con un elemento DOM.
<code>wrapInner(html)</code>	Envuelve el contenido secundario interno de cada elemento coincidente (incluidos los nodos de texto) con una estructura HTML.

Ejemplos

```
<p>Click on the square below:</p>
<span id = "result"> </span>

<div id = "division">
    This is Blue Square!!
</div>
```

```
$( "div" ).click(function () {
    var content = $(this).html();
    $( "#result" ).text( content );
});
```

```
$( "div" ).click(function () {
    $(this).replaceWith("<h1>JQuery !!</h1>");
});
```

```
$( "div" ).click(function () {
    $(this).remove();
});
```

```
$( "div" ).click(function () {
    $(this).before('<div class="div"></div>');
});
```

Funciones jQuery

- .each(): Itera sobre cada elemento
- .size(): Devuelve la cantidad de elementos
- .get(n): Obtiene el elemento n (:nth)
- .slice(n,m): Obtiene desde n hasta m-1
- .not('p'): No incluye los elementos p
- .add('p'): Agrega el elemento p
- .remove(): Elimina todos los elementos del DOM
- .empty(): Elimina el contenido de todos los elementos



Pongámoslo en práctica

- En la página itera sobre cada campo del formulario para obtener sus propiedades. Muéstralos concatenados en una tercera sección con título “Atributos de inputs”
- Añade bootstrap al proyecto. Rodea cada input por un div siguiendo el esquema de bootstrap (class="form-group")
- Añade un label a todos los campos en función de su atributo name
- Para el formulario, busca todos los inputs de tipo checkbox y remárcalos mediante css.

4

EVENTOS

Eventos

- Podemos crear páginas web dinámicas mediante el uso de eventos.
- Los eventos son acciones que pueden ser detectadas por una aplicación Web.
- Por ejemplo:
 - Un clic del ratón
 - Carga de una página web
 - Pasar el mouse sobre un elemento
 - Envío de un formulario HTML
 - Una pulsación en el teclado
 - Etc.
- Cuando se activan estos eventos, puede utilizar una función personalizada para reaccionar al evento.
- Estas funciones personalizadas llaman a los **controladores de eventos (Event Handlers)**

Binding de eventos

- Usando el modelo de eventos jQuery, podemos establecer manejadores de eventos en elementos DOM con el método bind().
- El método bind sigue la siguiente sintaxis:

selector.bind(eventType[, eventData], handler)

- **EventType**: cadena que contiene un tipo de evento JavaScript, como hacer clic o send.
- **EventData**: parámetro opcional es un mapa de datos que será pasado al controlador de eventos.
- **Handler**: función que se llama cada vez que se desencadena el evento.

```
<div class = "div">UNO</div>
<div class = "div">DOS</div>
<div class = "div">TRES</div>
```

```
$('.div').bind('click', function( event ){
    alert('Hola!!');
});
```

Quitar los handler de eventos

- Normalmente, una vez que un hanlder de eventos se establece, permanece durante el resto de la vida de la página.
- JQuery proporciona el comando **unbind()** para eliminar un handler de eventos.

```
selector.unbind(eventType, [handler] );
```

Atributos de evento

- La función de callback toma un parámetro cuando se llama al handler: el objeto de evento.
- Mediante dicho parámetro podemos accede a los atributos del evento

```
$('div').bind('click', function( event ){  
    alert('Event type is ' + event.type);  
    alert('pageX : ' + event.pageX);  
    alert('pageY : ' + event.pageY);  
    alert('Target : ' + event.target.innerHTML);  
});
```

Lista de métodos de eventos

Método	Descripción
preventDefault()	Evita que el navegador ejecute la acción predeterminada.
isDefaultPrevented()	Indica si event.preventDefault() se llamó alguna vez por este objeto de evento.
stopPropagation()	Detiene el event bubble de los elementos primarios, evitando que los controladores de los padres sean notificados del evento.
isPropagationStopped()	Indica si event.stopPropagation() se ha llamado alguna vez en este objeto de evento.
stopImmediatePropagation()	Detiene el resto de los hanlders de ejecución.
isImmediatePropagationStopped()	Indica si event.stopImmediatePropagation() se llamó alguna vez en este objeto de evento.

Métodos de manipulación de eventos

Método	Descripción
bind(type, [data], fn)	Vincula un handler a uno o más eventos (como clic) para cada elemento coincidente. También puede enlazar eventos personalizados.
off(events [, selector] [, handler(eventObject)])	Elimina un evento vinculado.
hover(over, out)	Simula el hovering, por ejemplo moviendo el ratón y desactivando un objeto.
on(events [, selector] [, data], handler)	Vincula un controlador a un evento (como clic) para todos los elementos actuales y futuros. También puede enlazar eventos personalizados.
one(type, [data], fn)	Vincula un handler a uno o más eventos que se ejecutarán una vez para cada elemento coincidente.
ready(fn)	Vincula una función que se ejecutará siempre que el DOM esté listo para ser recorrido y manipulado.
trigger(event, [data])	Dispara un evento en cada elemento coincidente.
triggerHandler(event, [data])	Inicia todos los controladores de eventos enlazados en un elemento.
unbind([type], [fn])	Elimina eventos enlazados de cada uno de los elementos emparejados.

Eventos de jQuery – Atajos de Binding

JQuery provee algunos atajos para eventos más comunes:

- .click(f): evento click
- .submit(f): evento submit
- .dblclick(f): evento double click
- .mouseover(f): evento mouse over
- .mouseout(f): evento mouse out
- .select(f): evento select

Propiedades del elemento this

- this: El objeto en sí
- this.id: El id del objeto
- this.tag: Nombre del objeto
- this.attr: Un atributo
- this.src: El src en links o includes
- this.classname: Nombre de una clase
- this.title: El título
- this.alt: El mensaje alt
- this.value: El valor en un elemento de formulario



Pongámoslo en práctica

- Asocia un evento al botón de envío del formulario que compruebe si todos los campos tienen valor y si no aborte el envío
- Añade un handler que cuando pases el ratón por la sección del artículo, la escurezca y cuando salgas, vuelva su estado previo.
- Haz que cuando hagamos clic en alguno de los elementos de la lista, se muestre su contenido en un alert.

5

Efectos

Efectos en jQuery

- › JQuery proporciona una interfaz simple para hacer varios tipos de efectos asombrosos.
- › Los métodos jQuery nos permiten aplicar rápidamente efectos de uso común con una configuración mínima.
- › Por ejemplo: ocultar o mostrar un elemento

```
<div class = "mydiv">  
    This is a SQUARE  
</div>  
  
<button id = "hide">ocultar</button>  
<button id = "show">mostrar</button>
```

```
$("#show").click(function () {  
    $(".mydiv").show( 1000 );  
});  
  
$("#hide").click(function () {  
    $(".mydiv").hide( 1000 );  
});
```

Lista de métodos de eventos

Método	Descripción
animate(params, [duration, easing, callback])	Una función para realizar animaciones personalizadas.
fadeIn(speed, [callback])	Se funden en todos los elementos compatibles ajustando su opacidad y activando una devolución de llamada opcional una vez finalizada.
fadeOut(speed, [callback])	Desvanece todos los elementos coincidentes ajustando su opacidad a 0, ajustando la pantalla a "ninguno" y disparando una devolución de llamada opcional después de completarse.
fadeTo(speed, opacity, callback)	Desvanece la opacidad de todos los elementos compatibles con una opacidad especificada y activar una devolución de llamada opcional una vez finalizada.
hide()	Oculta cada uno de los conjuntos de elementos coincidentes si se muestran.
hide(speed, [callback])	Oculta todos los elementos coincidentes con una animación elegante y disparar una devolución de llamada opcional después de la finalización.
show()	Muestra cada uno de los conjuntos de elementos coincidentes si están ocultos.
show(speed, [callback])	Muestra todos los elementos coincidentes mediante una animación elegante y disparar una devolución de llamada opcional después de la finalización.

Lista de métodos de eventos (II)

Método	Descripción
slideDown(speed, [callback])	Revela todos los elementos coincidentes ajustando su altura y disparando una devolución de llamada opcional después de la finalización.
1slideToggle(speed, [callback])	Cambia la visibilidad de todos los elementos coincidentes ajustando su altura y activando una devolución de llamada opcional después de la finalización.
1slideUp(speed, [callback])	Oculta todos los elementos coincidentes ajustando su altura y disparando una devolución de llamada opcional después de la finalización.
1stop([clearQueue, gotoEnd])	Detiene todas las animaciones en ejecución en todos los elementos especificados.
1toggle()	Alberna mostrando cada uno de los conjuntos de elementos coincidentes.
1toggle(speed, [callback])	Activa o desactiva la visualización de cada uno de los elementos combinados mediante una animación elegante y activa una devolución de llamada opcional una vez finalizada.
1toggle(switch)	Activa o desactiva la visualización de cada uno de los conjuntos de elementos coincidentes basados en el switch (true muestra todos los elementos, false oculta todos los elementos).
1jQuery.fx.off	Desactiva globalmente todas las animaciones.

Ejemplo

```
<div class = "content">
    <div class = "clickme">Click Me</div>
    <div class = "target">
        <img src = "./images/jquery.jpg" alt = "jQuery" />
    </div>
    <div class = "log"></div>
</div>
```

```
$(".clickme").click(function(event){
    $(".target").toggle('slow', function(){
        $(".log").text('Transition Complete');
    });
});
```



Pongámoslo en práctica

- Asocia un evento al botón de envío del formulario, además de la comprobación de valor; si es correcto haga que se desvanezca.

6

VALIDACIÓN DE FORMULARIOS

JQuery Validate

- Uno de los plugins más potentes para validar formularios.
- Está disponible en: <https://jqueryvalidation.org/>
- Para usarlo, es necesario añadir la referencia en el html

```
<script src="vendor/jquery-validation/dist/jquery.validate.min.js"></script>
```

Formulario + reglas de validación + mensajes

- Una vez creado el formulario

```
<form action="" name="registration">  
  
  <label for="email">Email</label>  
  <input type="email" name="email" id="email"/>  
  
  <button type="submit">Register</button>  
  
</form>
```

- Será necesario añadir las reglas de validación

.... (siguiente página >)

Formulario + reglas de validación (cont.)

```
$(function() {  
    // Initializa la validación del formulario con nombre "registration"  
    $("form[name='registration']").validate({  
        // Especifica las reglas de validación  
        rules: {  
  
            email: {  
                required: true,  
                email: true // Indica que es de tipo email  
            },  
  
        },  
        // Especifica los mensajes de error  
        messages: {  
            email: "Please enter a valid email address"  
        },  
        // Envía el formulario cuando es correcto  
        submitHandler: function(form) {  
            form.submit();  
        }  
    });  
});
```

Reglas de validación

- Podremos indicar las reglas a través de un objeto javascript
 - rules: {...},
- Dentro indicaremos por cada campo (por name) las reglas.
 - lastname: "required",
- Puede haber más de un regla por cada campo

```
password: {  
    required: true,  
    minlength: 5  
}
```

Reglas de validación (II)

- Para los campos que queramos sean obligatorios:
 - required: true
- Características que deberán cumplirse para ser validados
 - **maxlength**: comprobará que el número de caracteres es menor al especificado
 - **minlength**: comprobará que el número de caracteres es mayor al especificado
 - **rangelength**: comprobará que el número de caracteres introducidos se encuentra entre el rango especificado, por ejemplo rangelength:[50,250]
 - **min**: se aplica a campos numéricos, comprueba que como mínimo se introduzca un valor igual al especificado
 - **max**: se aplica a campos numéricos, comprueba que como mucho se introduzca un valor igual al especificado
 - **equalTo**: se usa para comprobar que el valor de dos campos sea el mismo, por ejemplo la confirmación de una contraseña o correo. Su uso sería de la siguiente forma: confirm_email:{equalTo:"#email"}
 - **url**: comprobará que el valor introducido tiene formato de URL
 - **email**: comprobará que el valor introducido tiene formato de e-mail

Reglas de validación (III)

➤ Validaciones condicionales

```
email:{  
    required:{  
        depends:"input[name='privacy-policy[]']:checked"  
    }  
}
```

Mensajes de error

- Podremos añadir mensajes para indicar al usuario de un error.
 - messages: {...},
- Para cada campo, según la regla que no se cumpla, podremos definir los mensajes:

```
messages: {  
    firstname: "Please enter your firstname",  
    lastname: "Please enter your lastname",  
    password: {  
        required: "Please provide a password",  
        minlength: "Your password must be at least 5 characters long"  
    },  
    email: "Please enter a valid email address"  
},
```



Pongámoslo en práctica

- Integra el plugin para la validación completa del formulario de alta.



BananaTube: Decisión en equipo

- Discute en equipo la conveniencia de usar jquery, respecto a usar javascript plano en BananaTube
- Tomad nota de las fortalezas y debilidades que hayas observado en cada caso
- Implementa la solución en función de la decisión



[...]**netmind**

WeKnowIT

Barcelona

C. Almogàvers, 123
08018 Barcelona
Tel. 93 304.17.20
Fax. 93 304.17.22

Madrid

Plaza Carlos Trías Bertrán, 7
28020 Madrid
Tel. 91 442.77.03
Fax. 91 442.77.07

www.netmind.es



GOBIERNO
DE ESPAÑA

MINISTERIO
DE ENERGÍA, TURISMO
Y AGENDA DIGITAL

red.es



ESTRATEGIA DE
EMPRENDIMIENTO Y
EMPLEO JUVENIL
garantía juvenil



Agenda Digital para España



UNIÓN EUROPEA

Fondo Social Europeo
"El FSE invierte en tu futuro"