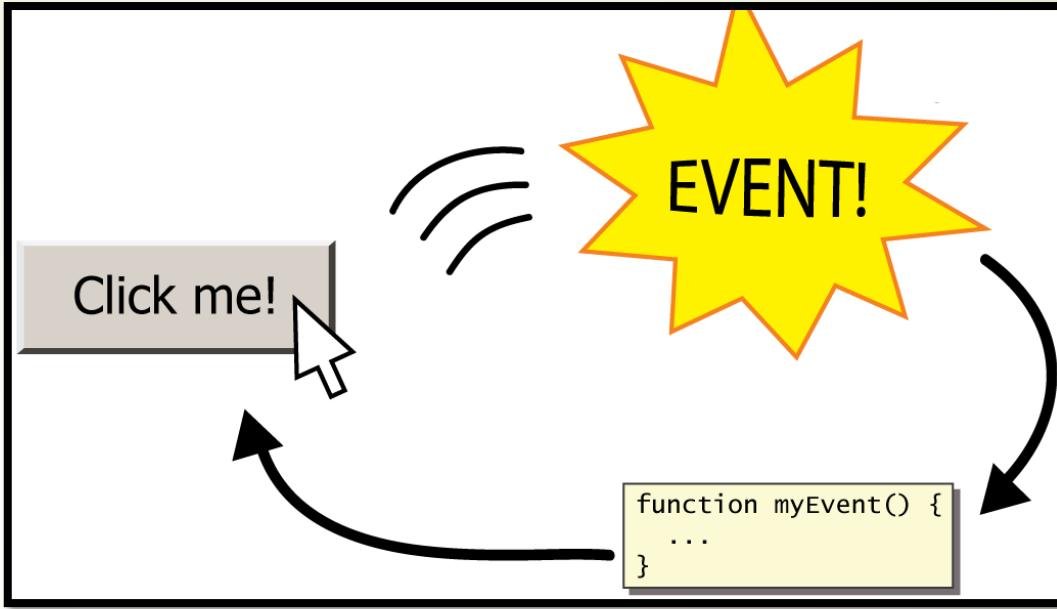


# Desarrollo de Aplicaciones Web

DOCENTE: Daniel López Lozano





# Tema 3.

## Interacción con el usuario.

## Gestión de Eventos

# Índice de contenidos

- Definición de evento en JavaScript.
- Introducción a JQuery.
  - Qué es JQuery.
  - DOM básico.
  - Gestión de eventos.

- Un evento es el desencadenante de una acción.
- Es lo que no permite interactuar con una aplicación web.
- Hasta ahora nuestras interacciones eran a través del alert, prompt y confirm.
- Otras formas son también los enlaces y el botón de enviar de un formulario.
- La forma esencial es el clic del ratón pero hay muchas más.

# ¿Qué eventos has observado en una aplicación, web o no, a lo largo de tu vida?

The image shows a digital task board interface. At the top, there's a header with icons for notifications, users, a search bar ("Search Everything..."), a megaphone, "Invite Team Members", and a user profile. A message says "This board is public, visible to all team members." Below the header, the board title is "Team Weekly Tasks" with a "Subscribe" button and a follower count "(1)". There's also a "Search/Filter Board" input field. The main area displays a grid of tasks under the heading "This Week".

This Week	Lead	Status	Timeline	Numbers	Priority	+
This is a pulse		Done	Apr 3 - 4	5	High	
A pulse can be anything you want it to be		Working on it	Apr 4 - 6	2	Low	
a task		Done	Apr 3 - 4	1	Low	
or a client		Working on it	Apr 10 - 13	4	High	
a project		Done	Apr 10 - 13	1	Low	

# Lista de eventos

Nombre	Causa
click	Click sobre un elemento
dblclick	Doble click sobre un elemento
mousedown	Se pulsa un botón del ratón sobre un elemento
mouseenter	El puntero del ratón entra en el área de un elemento
mouseleave	El puntero del ratón sale del área de un elemento
mousemove	El puntero del ratón se está moviendo sobre el área de un elemento
mouseover	El puntero del ratón se sitúa encima de cualquier elemento
mouseout	El puntero del ratón sale fuera de cualquier elemento
mouseup	Un botón del ratón se libera estando sobre un elemento
keydown	El usuario tiene pulsada una tecla
keypress	El usuario pulsa una tecla (momento justo en que la pulsa)
keyup	El usuario libera una tecla que tenía pulsada
scroll	El usuario usa el scroll vertical o horizontal

# Lista de eventos

Nombre	Causa
contextmenu	Cuando el usuario pulsa el botón derecho
focus	Un elemento del formulario toma el foco
blur	Un elemento del formulario pierde el foco
change	Un elemento del formulario cambia
load	Se ha completado la carga de la ventana
drag	Se arrastra un elemento por la web
drop	Se suelta un elemento en la web
input	Cambia un elemento de tipo input
select	Se selecciona con el ratón texto dentro de un input
submit	Cuando un formulario es enviado al servidor
beforeunload	Antes de salir de una página web

- ❑ Podríamos entender los eventos como bucles que esperan constantemente a que suceda dicho evento.
- ❑ El objetivo del tratamiento de eventos es realizar una acción cuando se produzcan.
- ❑ De forma general dichas acciones consisten en:
  - ✓ Seleccionar elementos y cambiarlos.
  - ✓ Agregar nuevo contenido.
  - ✓ Ocultar y mostrar elementos en la web
  - ✓ Validación de datos de entrada.
  - ✓ Efectos visuales vistosos.

- Existen 3 formas actualmente de poder usar eventos.
- Incrustado en un etiqueta, eventos semánticos y estándar de la W3C.
- Vamos a aplicarlo a un ejemplo simple con imágenes.



# Incrustado o modelo tradicional

```
<body>
  
```

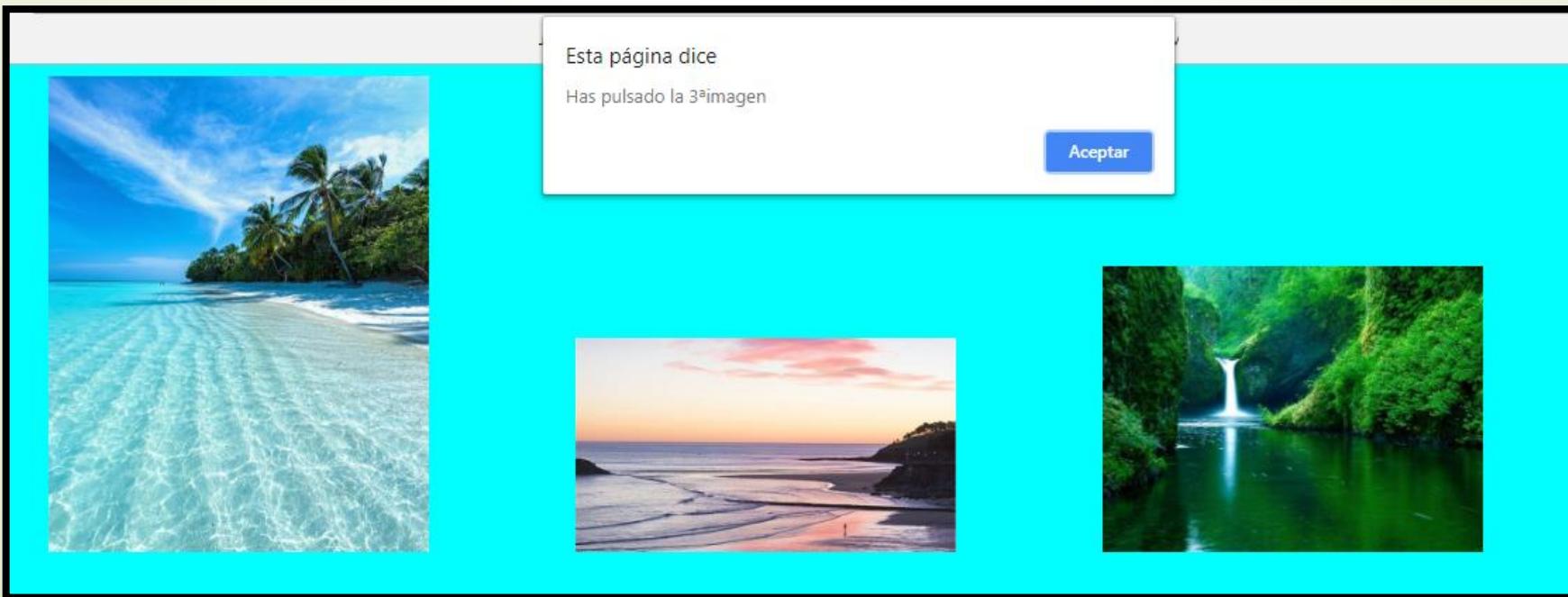
## Eventos semánticos

```
window.onload=Inicio;
function Inicio()
{
  alert("La pagina ya esta cargada");
}

//Usando funciones anonimas
window.onload=function()
{
  alert("La pagina ya esta cargada");
};
```

# Eventos W3C

```
document.addEventListener("click",Cuerpo);
function Cuerpo()
{
    document.body.style.backgroundColor="cyan";
}
//Usando funciones anonimas
document.addEventListener("click",
function()
{
    document.body.style.backgroundColor="cyan";
});
```



- ❑ JQuery es una librería que reduce considerablemente el código escrito y ofrece una gran cantidad de problemas resueltos para no tener que programarlas una y otra vez.
- ❑ Sobre esta librería existen extensiones (plugins) que se pueden incorporar (JQuery validation, JQueryUI, JQueryMobile, etc) para hacer más completas y mas fáciles de programar nuestras aplicaciones web.
- ❑ JQuery no es algo diferente a JavaScript ya que son un conjunto de definiciones nuevas para hacer lo mismo.

- En un principio JQuery esta ampliamente soportado y en teoría no debería quedarse anticuado, aunque los puristas de JavaScript no les agrada mucho.
- Es interesante no depender de JQuery y en primer lugar tener una buena base en JavaScript.
- Para usar JQuery hay varias opciones donde la primera decisión es si la descargamos o usamos un CDN (Content Delivery Network) para enlazarla desde Internet.
- A nivel de código no existe ninguna diferencia.

- Existen diversos servidores de otras empresas que mantienen versiones de la librería, además del oficial de JQuery, como Microsoft y Google.
- Si queremos descargarla debemos dirigirnos a <https://jquery.com/download/> y descargamos la versión compressed (minificada) donde los espacios, saltos de línea y tabulaciones innecesarias están eliminadas y por tanto ocupa menos.
- Si optamos por usar un CDN, el oficial de JQuery es <https://code.jquery.com/> donde obtendremos un código HTML que deberemos copiar en nuestra aplicación.

# Inclusión de JQuery y comparación con JavaScript

```
<script type="text/javascript" src="jquery-3.2.1.min.js"></script>
<script type="text/javascript" src="app.js"></script>
```

//JavaScript Puro	//JQuery
document.addEventListener("load",Inicio);	\$(document).ready(Inicio);
document.addEventListener("click",Cuerpo);	\$("body").click(Cuerpo);
function Cuerpo()	function Cuerpo()
{	{
document.body.style.backgroundColor="cyan";	\$(body).css("backgroundcolor","cyan");
}	}
var imagenes=document.getElementByName("img");	
for (let i=0;imagenes.length;i++)	\$("img").click(Mensaje);
{	
imagenes.addEventListener("click",Mensaje);	
}	

# Más cosas interesante con JQuery

```
$("body").hide();
$("body").hide(3000);
//Encadenamiento de funciones o chaining
$("body").hide().show(3000);
$("body").hide().fadeIn(3000);
$("img").hide().width(300).height(500).css("border", "3px solid red").slideDown(3000);
```

Todo esto no deja de ser JavaScript !!!!

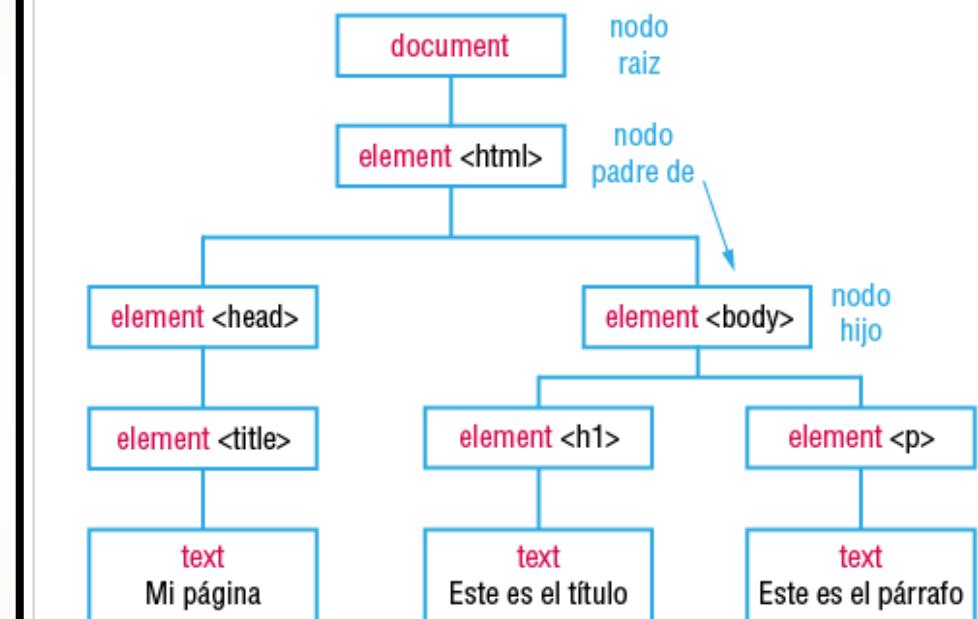
- El DOM (Modelo de objetos del documento) define la estructura lógica de los documentos HTML y el modo en que se accede y manipula un documento.
- Con el Modelo de Objetos del Documento los programadores pueden construir documentos, navegar por su estructura, y añadir, modificar o eliminar elementos y contenido.
- Cada etiqueta se simboliza mediante un nodo que es objeto desde el punto de vista la POO y esta compuesto de todas las características que pueda tener una etiqueta HTML.

# Estructura DOM

Documento HTML

```
<html>
  <head>
    <title> Mi página </title>
  </head>
  <body>
    <h1> Este es el título </h1>
    <p> Este es el párrafo </p>
  </body>
</html>
```

Presentación DOM



- El DOM proporciona una serie de funciones que permiten acceder a cualquier nodo de la página de forma sencilla e inmediata.
- En un principio el DOM se especificaba con un Array por cada etiqueta posible HTML
- Hoy día eso no tiene sentido debido al gran numero de elementos presentes en un pagina web.
- Se puede acceder por etiqueta, por id y por clase según el DOM actual.

- ❑ No podemos olvidar que para hacer cualquier operación con el DOM los elementos de la pagina deben estar cargados o listos.
- ❑ En otro caso podría resultar que un código correcto no funcione porque el elemento no existe.

```
$('document').ready(Inicio);

function Inicio()
{
    /*Todo el codigo necesario para
     nuestra pagina web*/
}
```

# Funciones DOM para acceder al HTML

```
let encabezado=document.getElementByName("h1");
let id_noticias=document.getElementById("noticias");
let articulos=document.getElementsByClassName("articulo");
```

Es deducible que JQuery nos facilitara las cosas

```
Let encabezado=$("#h1");
Let id_noticias=$("#noticias");
Let articulos$(".articulo");
```

Para saber el tipo de los elementos seleccionados

```
alert($("#principal").prop("tagName"));
```

# Vemos que JQuery se basa en selectores CSS

```
//Podemos guardarLo en una variable  
//si vamos a utilizarlo más veces  
$(".clear");  
$("#menu_principal");  
$("div");
```

Podemos hacer las mismas combinaciones CSS

```
$("#principal a");  
$("table .subrayado");  
$("div.articulo");  
$("p,h1,span");  
$("input[type=submit]");
```

- Existen funciones para ver y modificar el contenido de los elementos de la pagina.

```
$("div#principal").text(); //visualiza el texto del elemento seleccionado con $  
$("div#principal").text("Texto nuevo"); //modifica el texto del elemento seleccionado  
  
$("div#principal").html(); //visualiza el HTML del elemento seleccionado con $  
$("div#principal").html("<h1>HTML nuevo</h1>"); //modifica el HTML del elemento seleccionado
```

- También disponemos de las dimensiones y posición.

```
//Informacion dimensional del elemento boton  
let caja=$("#boton");  
alert("Ancho: "+caja.width()+" Alto: "+caja.height());  
let coordenadas=caja.position();  
alert("Top position: " + coordenadas.top + " Left position: " + coordenadas.left)  
  
//Para modificar Las dimensiones  
caja.width(300);  
caja.height(500);  
caja.offset({top:10,left:40});
```

# Modificar propiedades CSS

```
//Todos Los parrafos a color rosa
$("p").css("background-color","pink");

//Todos Los div con clase articulo a fuente 2em
$("div.articulo").css("font-size","2em");

//Varias propiedades en un sola Linea
$("p").css({"background-color": "yellow", "font-size": "200%"});

//Si Las propiedades son numericas
$("p").css( "border-width", "+=3" );
```

- La función CSS nos devuelve el valor actual de la propiedad.

```
$("#boton").css("font-family");
```

- Cuando la propiedad es un color devuelve una cadena con el RGB. Hay que fijarse en los espacios.

```
alert($("#boton").css("background-color"));
```

Una página insertada en esta dice

rgb(221, 221, 221)

Aceptar

- Si la propiedad es numérica hay que aplicar parseInt si queremos realizar una comparación.

```
let texto=$("#secundario");
if(parseInt(texto.css("font-size"))>20)
{
    alert("El texto es demasiado grande");
}else{
    texto.css("font-size","+=1");
}
```

- Algunas veces es engoroso trabajar con varias propiedades por lo que se suele agrupar en clases CSS.

- Podemos manejar las clases CSS asociadas a un elemento de nuestra web.

```
let boton_ejemplo=$("#boton");
boton_ejemplo.addClass("boton_personalizado");
boton_ejemplo.removeClass("boton_personalizado");
//Efecto interruptor
if(boton_ejemplo.hasClass("boton_personalizado"))
{
    boton_ejemplo.removeClass("boton_personalizado");
}else{
    boton_ejemplo.addClass("boton_personalizado");
}
//JQuery sigue simplificandonos el trabajo
boton_ejemplo.toggleClass("boton_personalizado");
```

- Con la función attr podemos tomar el valor o modificar un atributo del elemento seleccionado.

```
//Ver un atributo          //Modificar un atributo
$("a").attr("href");
$("img").attr("src");
$("table").attr("border");
```

```
$( "a" ).attr( "href" , "ejemploDOM.html" );
$( "img" ).attr( "src" , "encendida.gif" );
$( "table" ).attr( "border" , "2" );
```

- Para atributos fijos como disabled, required, etc se utiliza la función removeAttr

```
//Habilitamos un boton porque el formulario es correcto
//Le damos el foco para que se active al pulsar intro
$("#boton").removeAttr("disabled").focus();
```

- Ya hemos utilizado animación básica de JQuery, de la que existen 3 tipos.

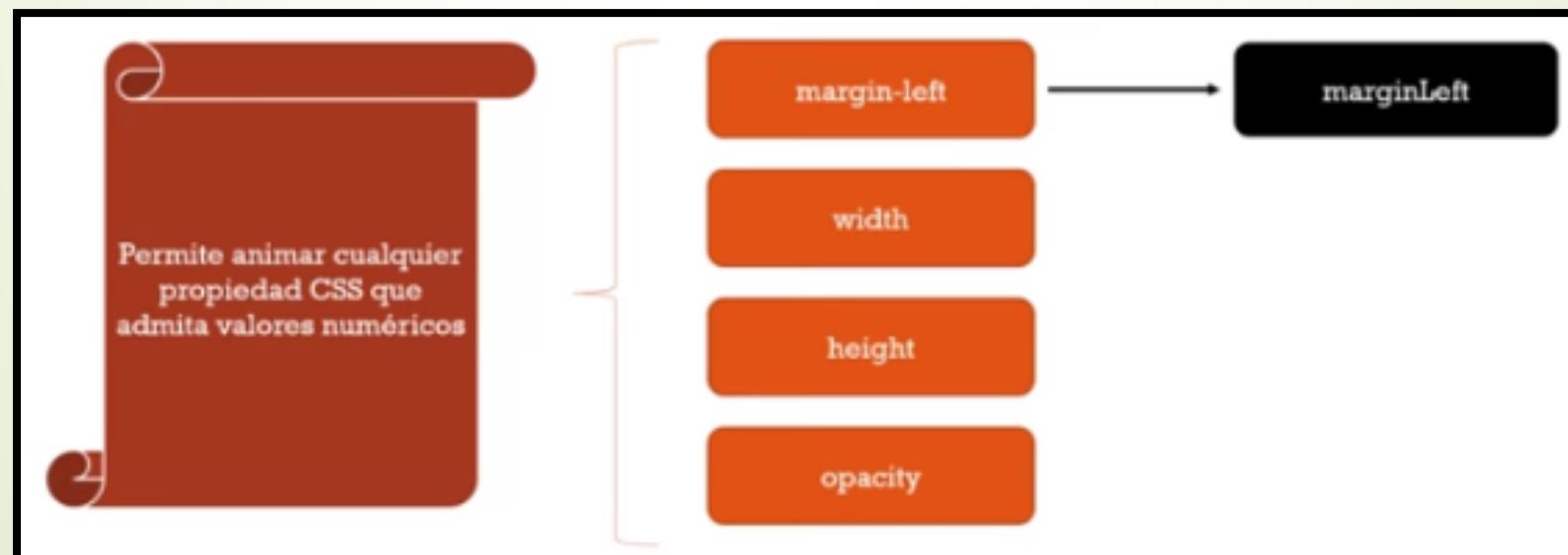
```
$("div#principal").hide();
$("div#principal").show();
$("div#principal").toggle();

$("div#principal").fadeOut(1000);
$("div#principal").fadeIn();
$("div#principal").fadeToggle();

$("div#principal").slideUp();
$("div#principal").slideDown(
    function(){
        alert("Se ha terminado la animacion");
    });
$("div#principal").slideToggle();
```

- Con la función animate se crea un efecto personalizado.

```
$("div").animate({
    left: '250px',
    opacity: '0.5',
    height: '150px',
    width: '150px'
});
```



- ❑ ¿Qué hace el siguiente código en JQuery?

```
$("img").fadeOut(1000).fadeIn(1000).width(300);
```

- ❑ No siempre queremos aplicar una función a la selección completa. JQuery nos proporciona funciones para filtrar colecciones de objetos.

```
var primero=$("p").first();
var ultimo=$("p").last();
var tercero=$("p").eq(2); //empieza en cero
```

- ❑ Es necesario en algunos casos procesar objeto a objeto una colección para filtrar o extraer información de los mismos

- Imaginemos una web con enlaces a diversas páginas.
- Es interesante hacer un bibliografía de enlaces, pero sin tener que apuntar uno a uno y que al añadir más se actualice automáticamente.

```
let direcciones="";
let enlaces=$("a");
for(let i=0;i<enlaces.length;i++)
{
    direcciones+="

Enlace: "+i+" "+enlaces.eq(i).attr("href")+"

";
}
//append añade contenido al final de una etiqueta
$("body").append("Bibliografia: "+direcciones);
```

- ❑ JQuery tiene una construcción llamada each que nos evita el hecho de usar bucles aunque el código es más difícil de entender.

```
Let direcciones="";
$("a").each(
    function(i,enlace)
    {
        direcciones+="

Enlace: "+i+" "+enlace.attr("href")+"

";
    });
$("body").append("Bibliografia: "+direcciones);
```

- ❑ No es muy recomendable aunque hay manuales que la usan mucho.

- Hasta ahora solo hemos visto el evento de clic y de carga(ready).
- Desde JQuery podemos gestionar eventos mediante funciones con su nombre.

```
$("p").click(...);  
$("p").dblclick(...);  
$("p").mouseenter(...);  
$("p").mouseleave(...);  
$("p").mousemove(...);
```

- Aunque no todos los eventos tienen una, por lo que existe la función genérica on

```
$("p").on("click",Pulsar);
```

- El clic y el mouseenter junto con mouseleave son de los eventos más utilizados en una aplicación.
- Por ello JQuery añade 2 gestores de eventos especiales relacionados con ellos.
- Se trata de hover y toggle.
- Son especiales porque necesitan dos funciones para programarse correctamente.
- El más simple hover controla la entrada y salida del ratón en un solo evento.

- Vamos a hacer un efecto rollover con dos imágenes comparando los dos mecanismos.

```
$("img#portada").mouseenter(CambiaImagen);
$("img#portada").mouseleave(VolverImagen);
//Usando hover
$("#img#portada").hover(CambiaImagen,VolverImagen);

function CambiaImagen()
{
    $("img#portada").attr("src","paisaje2.jpg");
}

function VolverImagen()
{
    $("img#portada").attr("src","paisaje1.jpg");
}
```

- El evento toggle es más especial todavía porque permite crear un efecto on/off sobre un elemento de la pagina sin necesidad de un if-else.



```
//Sin toggle
$("img#bombilla").click(ControlBombilla);
function ControlBombilla()
{
    let imagen=$("img#bombilla");
    if(imagen.attr("src")=="apagada.gif")
    {
        imagen.attr("src","encendida.gif");
    }else{
        imagen.attr("src","apagada.gif");
    }
}
```



```
//Con toggle
$("img#bombilla").click(Apagar,Encender);
function Apagar()
{
    ...
    imagen.attr("src","apagada.gif");
}

function Encender()
{
    ...
    imagen.attr("src","encendida.gif");
}
```

- Resulta muy interesante aunque puede que se os olvide el if-else.

- ❑ Otro evento especial es el scroll, tanto en la forma de asignar una función al evento como la lectura del estado en que se encuentra el scroll.
- ❑ Es necesario para conseguir los típicos menús fijos (sticky) y el archiconocido efecto parallax.

```
$document).scroll(CambiarColor);
function colorScroll()
{
    if ($document.scrollTop() > 50) {
        $("body").css("background-color","red");
    }else{
        $("body").css("background-color","inherit");
    }
}
```

- El resto de eventos no vistos que tiene que ver con el teclado son más típicos en los formularios.
- Para acceder a la información que contiene un elemento de formulario HTML no se usa la función HTML si no la función val.
- Eventos muy relacionados a los formulario son change, blur, input, keypress , etc.
- Usados sobre todo para evitar valores inválidos en un formulario conforme lo escribimos.

- Los siguientes códigos controlan el contenido de un formulario como el siguiente:

Nombre:

Enviar al servidor

```
$("#nombre").blur(Vacio);
function Vacio()
{
    if($("#nombre").val() == "")
    {
        alert("Rellena tu nombre");
    }
}

$("#nombre").keyup();
function Longitud()
{
    let nombre=$("#nombre").val()
    if(nombre.length < 5)
    {
        alert("Al menos 5 caracteres");
    }
    if(nombre.length > 15)
    {
        alert("Maximo 15");
    }
}
```

- Cuando el formulario contiene fallos no debe enviarse al servidor.
- Para ello necesitamos un objeto de tipo event que hasta ahora lo habíamos ignorado.
- Dicho objeto contiene información acerca del evento producido y lo controla.
- La función preventDefault evita el comportamiento por defecto, en este caso el envío (submit) del formulario.

- ❑ No solo se usan en el submit de un formulario, también se usan para evitar la aparición del menú contextual y deshabilitar un hipervínculo (<a>).

```
$("body").contextmenu(EvitarBotonDerecho);
function EvitarBotonDerecho(evento)
{
    evento.preventDefault();
}
```

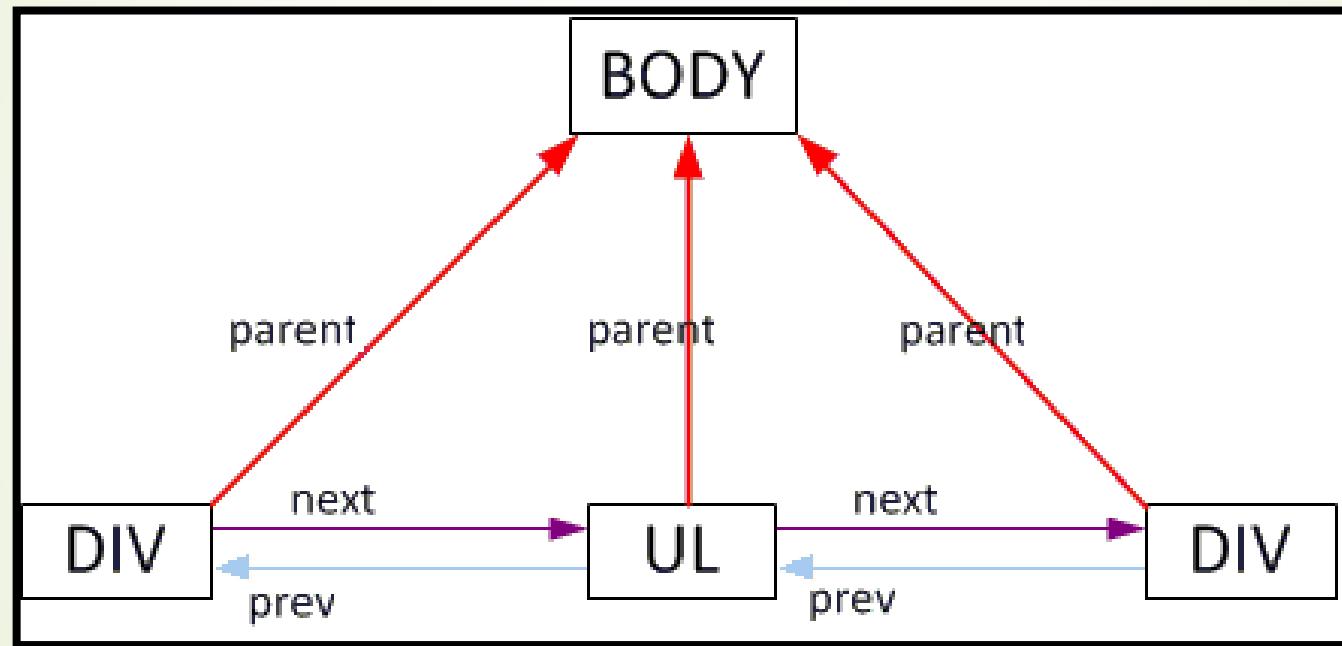
```
$("a").click(Confirmacion);
function Confirmacion(evento)
{
    if(!confirm("¿Desea salir de la web?"))
    {
        evento.preventDefault();
    }
}
```

- Por ultimo, el objeto event contiene información tanto de la tecla que se haya pulsado como de la posición del ratón en el momento del evento.

```
$("body").keyup(Info_tecla);
function Info_tecla(evento)
{
    document.write("Ha pulsado: "+String.fromCharCode(evento.which));
}

$("body").click(Info_raton)
function Info_raton(evento){
    document.write("Posicion: ("+evento.pageX+","+evento.pageY+ ")");
}
```

- ❑ JQuery nos permite hacer traversing en el DOM o lo que es lo mismo tener acceso directo a los nodos que rodean a uno en concreto.



- ❑ Dichas funciones son útiles en situaciones comunes.

- En los formularios a la derecha de cada input puede haber un span sin contenido.

Firstname  ✓ next()

Lastname  ✗ El apellidos es obligatorio

Password  ✓

Confirm password  ✗ Tienen que coincidir los password

Please agree to our policy  ✗ Tienes que estar de acuerdo con las condiciones

```
$("#apellidos").blur(Aviso);
function Aviso()
{
    let siguiente=$("#apellidos").next();
    siguiente.text("El apellido es obligatorio");
    ...
}
```

- Al pulsar el icono del cubo de basura hay que borrar es que esta su izquierda (prev).

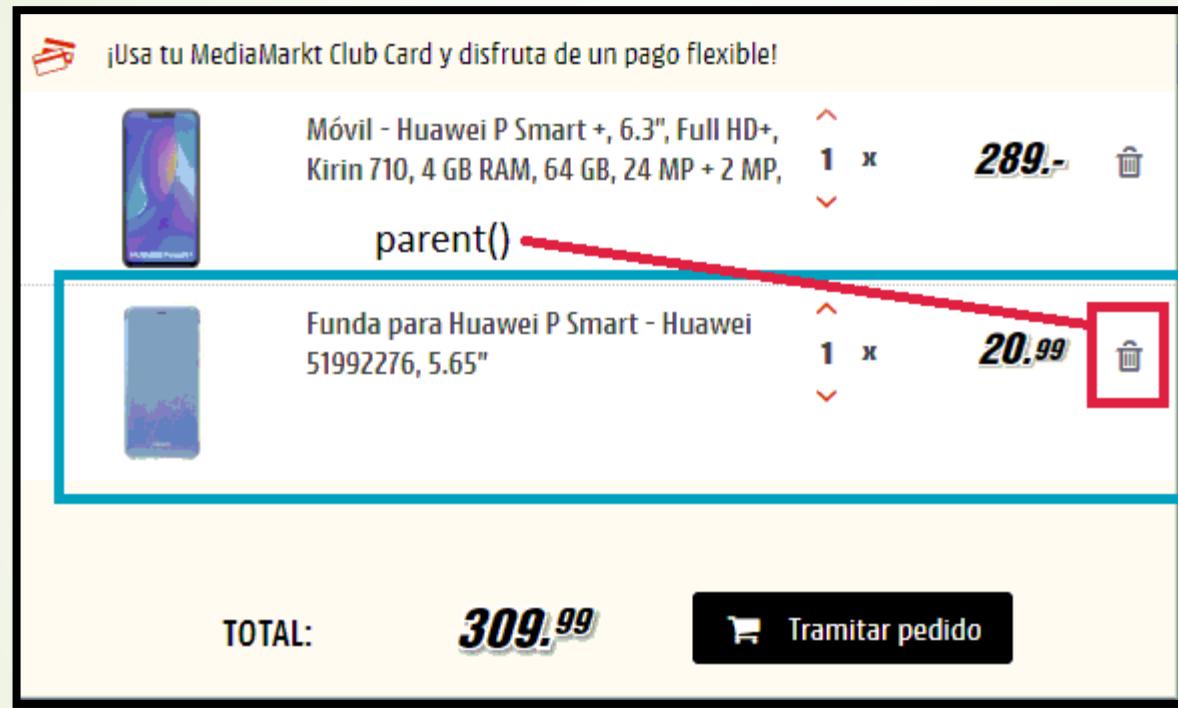
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed mattis enim vitae orci. Phasellus libero. Maecenas nisl arcu, posuere vitae, rutrum et, luctus at, pede. Pellentesque massa ante, ornare id, aliquam vitae, ultrices porttitor, pede. Nullam faucibus ut, rhoncus non, mi. Duis pellentesque, felis eu adipiscing ullamcorper, odio urna consequat arcu, at posuere ante scelerisque.

Ocultar contenidos

prev()

```
function Ocultar()
{
    $("a").prev().hide();
}
```

- Al pulsar el icono del cubo de basura hay que borrar ala su contenedor que es su elemento padre.



```
function Borrar()
{
    $("#rubbish-icon").parent().remove();
}
```

- ❑ Los eventos en JQuery se pueden aplicar de manera masiva.
- ❑ Eso esta bien si queremos que la respuesta sea exactamente la misma.
- ❑ En la mayoría de los casos la respuesta a un evento es un poco distinta dependiendo del objeto de la pagina que recibe el evento.
- ❑ No hay más que mirar atrás y la mayoría de los ejemplos hacen referencia al propio objeto que recibe la pagina.



- Si queremos que cada imagen nos diga su atributo src tenemos dos opciones.

```
//PRIMERA OPCION
$("#im1").click(Imagen1);
function Imagen1()
{
    alert($("#im1").attr("src"));
}
$("#im2").click(Imagen2);
function Imagen2()
{
    alert($("#im2").attr("src"));
}

//Y SI HAY 50 IMAGENES???
```

- ❑ Podríamos pensar en usar parámetros en las funciones y un bucle, pero es una técnica compleja que requiere de un conocimiento avanzado del ámbito de una función.
- ❑ Los lenguajes orientados a objetos tienen el concepto del this que es una referencia al propio objeto que recibe el evento.

```
//SEGUNDA OPCION
//Selector que coja un conjunto de imágenes
$("img").click(Info_img);
function Info_img()
{
    alert($(this).attr("src"));
}
```

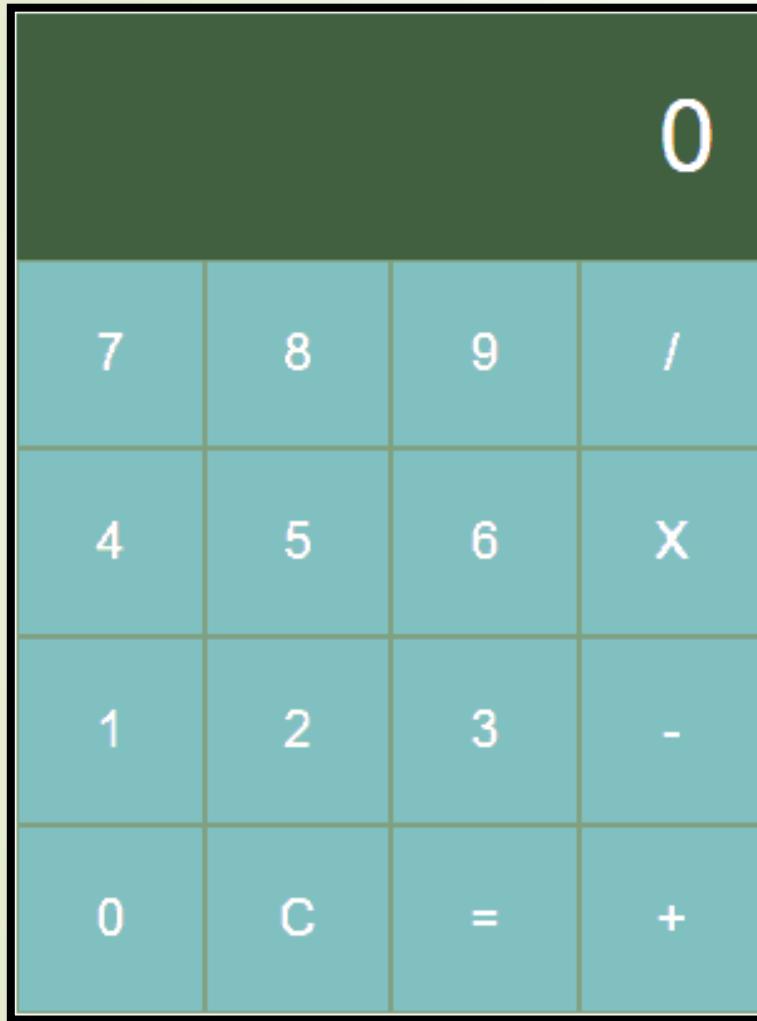
- ❑ Da igual que sean 1 o 1000 cada una se refiere a si misma (this) sin necesidad de repetirse.

- Es una técnica que no cambia nada de lo que hemos visto hasta ahora y nos ayuda a adaptar los códigos.

```
<a href="destino1.html">Enlace a destino1</a>
<a href="destino2.html">Enlace a destino2</a>
<a href="destino3.html">Enlace a destino3</a>
<a href="destino4.html">Enlace a destino4</a>
```

```
$( "a" ).click(Destino);
function Destino()
{
    alert($(this).attr("href"));
}
```

# ¿En que resultaría útil el this en estas apps?



¡Usa tu MediaMarkt Club Card y disfruta de un pago flexible!

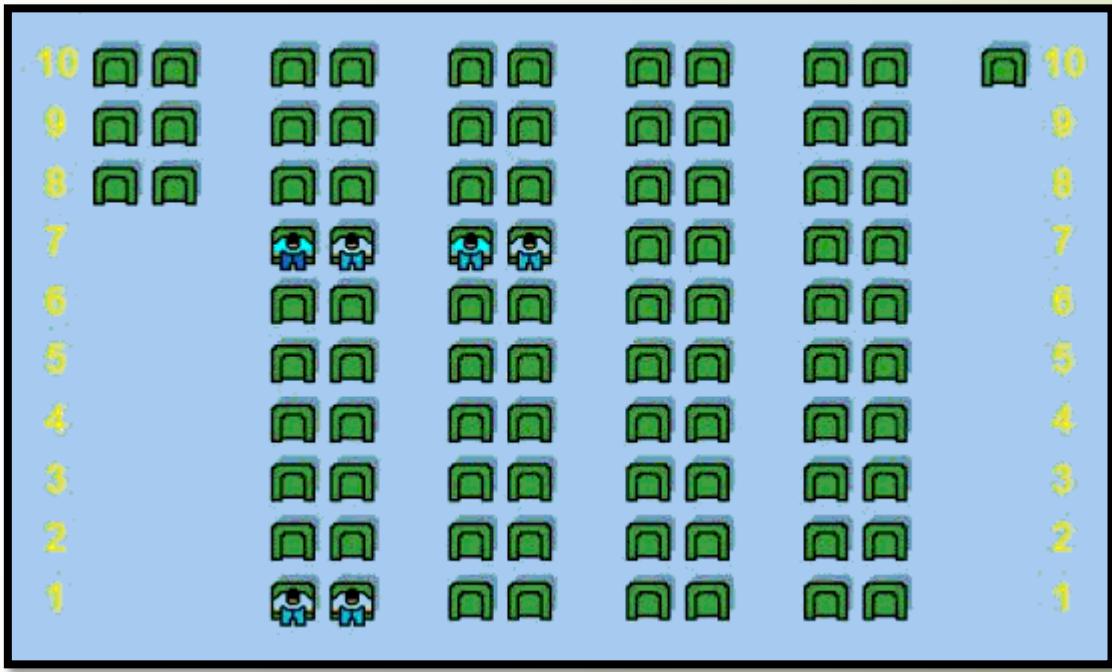
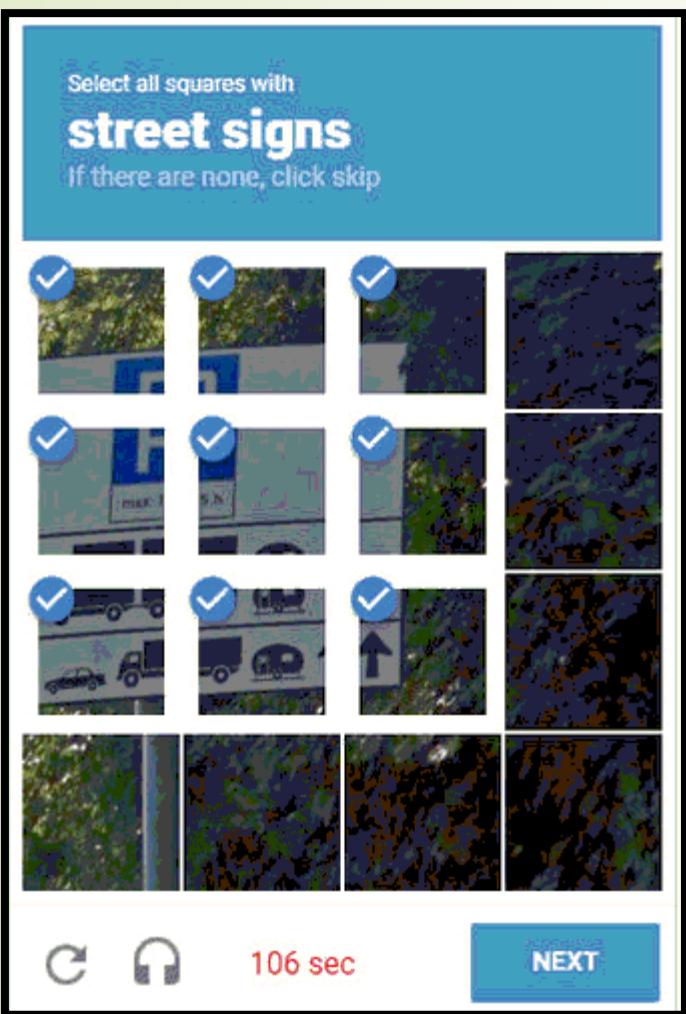
Móvil - Huawei P Smart +, 6.3", Full HD+, Kirin 710, 4 GB RAM, 64 GB, 24 MP + 2 MP, 1 x 289.-

Funda para Huawei P Smart - Huawei 51992276, 5.65"

TOTAL: 309.99

Tramitar pedido

## Otras situaciones donde el this salva la vida



# Bibliografía

- ❑ Gauchat, Juan Diego: “El gran libro de HTML5, CSS3 y Javascript”.  
Editorial Marcombo. 2012
- ❑ Vara, J.M. y otros: “Desarrollo Web en Entorno Cliente. CFGS”.  
Editorial Ra-Ma. 2012
- ❑ “Programación en Javascript”.  
Colección de artículos disponibles en la url.  
<http://www.desarrolloweb.com/manuales/> Última visita: Septiembre 2017.
- ❑ W3SCHOOL “Manual de referencia y Tutoriales”  
<http://www.w3schools.com/> Última visita Septiembre 2017
- ❑ Comesaña, J.L. : Técnico en Desarrollo de Aplicaciones Web.  
<http://www.sitiolibre.com/daw.php>  
Última visita Septiembre 2017
- ❑ Píldoras Informáticas. Curso de JavaScript.  
<https://www.pildorasinformaticas.es/course/javascript-desde-0/>  
Ultima visita Septiembre 2018.