Animaciones con JQuery

Mauro Bender

Mayo 31, 2012

JQuery

JQuery

- Es una librería en javascript que nos permite, de forma fácil, modificar los elementos del DOM.

JQuery

- Es una librería en javascript que nos permite, de forma fácil, modificar los elementos del DOM.
- Para obtener los elementos del DOM usamos selectores como los de css.

JQuery

- Es una librería en javascript que nos permite, de forma fácil, modificar los elementos del DOM.
- Para obtener los elementos del DOM usamos selectores como los de css.
- Es facilmente extensible y debido a eso existen gran variedad de plugins..

JQuery

- Es una librería en javascript que nos permite, de forma fácil, modificar los elementos del DOM.
- Para obtener los elementos del DOM usamos selectores como los de css.
- Es facilmente extensible y debido a eso existen gran variedad de plugins..
- Vamos a ver los como hacer animaciones con los elementos que nos proporciona esta librería.

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

```
<script type="text/javascript" language="javascript" src="
    jquery.js"></script>
```

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

```
<script type="text/javascript" language="javascript" src="
    jquery.js"></script>
```

Si no disponemos de una copia local del archivo que contiene la librería siempre podemos usar una copia que se encuentra alojada en los servidores de google con el siguiente código:

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

```
<script type="text/javascript" language="javascript" src="
    jquery.js"></script>
```

Si no disponemos de una copia local del archivo que contiene la librería siempre podemos usar una copia que se encuentra alojada en los servidores de google con el siguiente código:

```
<script type="text/javascript" language="javascript" src="
    https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/
    jquery.min.js"></script>
```

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

```
<script type="text/javascript" language="javascript" src="
    jquery.js"></script>
```

Si no disponemos de una copia local del archivo que contiene la librería siempre podemos usar una copia que se encuentra alojada en los servidores de google con el siguiente código:

```
<script type="text/javascript" language="javascript" src="
    https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/
    jquery.min.js"></script>
```

Esto es especialmente útil cuando no se pueden subir al servidor archivos javascript.

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

```
<script type="text/javascript" language="javascript" src="
    jquery.js"></script>
```

Si no disponemos de una copia local del archivo que contiene la librería siempre podemos usar una copia que se encuentra alojada en los servidores de google con el siguiente código:

```
<script type="text/javascript" language="javascript" src="
    https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/
    jquery.min.js"></script>
```

Esto es especialmente útil cuando no se pueden subir al servidor archivos javascript. Para que nuestro código funcione correctamente es importante que nuestro la página se cargue completamente antes de que empecemos a modificarla. Para eso usamos el evento **ready** del documento:

Para poder usar la librería debemos incluir el archivo javascript que la contiene. Si el archivo se encuentra en la misma carpeta que el archivo HTML, podemos usar:

```
<script type="text/javascript" language="javascript" src="
    jquery.js"></script>
```

Si no disponemos de una copia local del archivo que contiene la librería siempre podemos usar una copia que se encuentra alojada en los servidores de google con el siguiente código:

```
<script type="text/javascript" language="javascript" src="
    https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/
    jquery.min.js"></script>
```

Esto es especialmente útil cuando no se pueden subir al servidor archivos javascript. Para que nuestro código funcione correctamente es importante que nuestro la página se cargue completamente antes de que empecemos a modificarla. Para eso usamos el evento **ready** del documento:

```
$(document).ready(function() {
// Aca va a ir nuestro codigo...
});
```

Selectores

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**. $ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.
Por ejemplo si tenemos un div como:

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.
Por ejemplo si tenemos un div como:

```
<div id="box">Esto es una caja</div>
```

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.
Por ejemplo si tenemos un div como:

```
<div id="box">Esto es una caja</div>
```

Podemos obtenerlo con:

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.
Por ejemplo si tenemos un div como:

```
<div id="box">Esto es una caja</div>
```

Podemos obtenerlo con:

```
var elem = $("#box");
```

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.
Por ejemplo si tenemos un div como:

```
<div id="box">Esto es una caja</div>
```

Podemos obtenerlo con:

```
var elem = $("#box");
```

Si en cambio queremos obtener todos los elementos con la clase “item”, simplemente hacemos:

Selectores

Para acceder a los elementos del DOM y modificarlos vamos a usar los selectores que nos proporciona JQuery, como por ejemplo:

```
var $links = $("a.link");
```

Este selector selecciona **TODOS** los elementos **a** que tenga la clase **link**.
$ es un alias para la clase JQuery, y por lo tanto $() construye un nuevo objeto de la clase JQuery que contiene todos los elementos a los que hace referencia el selector pasado.
Para seleccionar los elementos podemos usar expresiones de CSS.
Por ejemplo si tenemos un div como:

```
<div id="box">Esto es una caja</div>
```

Podemos obtenerlo con:

```
var elem = $("#box");
```

Si en cambio queremos obtener todos los elementos con la clase “item”, simplemente hacemos:

```
var elems = $(".item");
```

Más ejemplos de selectores

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

```
var elems = $("div[data=username]");
```

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

```
var elems = $("div[data=username]");
```

Esto selecciona todos los *div*s que tengan el atributo *data* y este sea igual a *username*.

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

```
var elems = $("div[data=username]");
```

Esto selecciona todos los *div*s que tengan el atributo *data* y este sea igual a *username*.

```
var elems = $("*[seleccioname]");
```

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

```
var elems = $("div[data=username]");
```

Esto selecciona todos los *div*s que tengan el atributo *data* y este sea igual a *username*.

```
var elems = $("*[seleccioname]");
```

Esto selecciona todos los elementoss que tengan el atributo *seleccioname*.

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

```
var elems = $("div[data=username]");
```

Esto selecciona todos los *div*s que tengan el atributo *data* y este sea igual a *username*.

```
var elems = $("*[seleccioname]");
```

Esto selecciona todos los elementoss que tengan el atributo *seleccioname*.

```
var elems = $("a.clase-1, a.clase-2");
```

Más ejemplos de selectores

```
var elems = $("#lista > li");
```

Esto selecciona todos los *li*s de la lista con id *lista*.

```
var elems = $("#lista li:first");
```

Esto selecciona el primer *li* de la lista con id *lista*.

```
var elems = $("#lista li:last");
```

Esto selecciona el último *li* de la lista *#lista*.

```
var elems = $("div[data=username]");
```

Esto selecciona todos los *div*s que tengan el atributo *data* y este sea igual a *username*.

```
var elems = $("*[seleccioname]");
```

Esto selecciona todos los elementoss que tengan el atributo *seleccioname*.

```
var elems = $("a.clase-1, a.clase-2");
```

Esto selecciona todos los *a*s que tengan la clase *clase-1* y o la clase *clase-2*.

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente.

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente. Por ejemplo, si queremos seleccionar el primer *p* que aparece en el documento, podemos hacer:

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente. Por ejemplo, si queremos seleccionar el primer *p* que aparece en el documento, podemos hacer:

```
var elems = $("p:eq(0)");
```

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente. Por ejemplo, si queremos seleccionar el primer *p* que aparece en el documento, podemos hacer:

```
var elems = $("p:eq(0)");
```

Esto busca todos los *p*s del documento y se queda con el primero de ellos.

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente. Por ejemplo, si queremos seleccionar el primer *p* que aparece en el documento, podemos hacer:

```
var elems = $("p:eq(0)");
```

Esto busca todos los *p*s del documento y se queda con el primero de ellos. O si queremos todos *div*s que tengan la clase *cajita* y estén visibles, podemos usar:

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente. Por ejemplo, si queremos seleccionar el primer *p* que aparece en el documento, podemos hacer:

```
var elems = $("p:eq(0)");
```

Esto busca todos los *p*s del documento y se queda con el primero de ellos. O si queremos todos *div*s que tengan la clase *cajita* y estén visibles, podemos usar:

```
var elems = $("div.cajita:visible");
```

También disponemos de selectores que son propios de JQuery y se pueden combinar con los selectores CSS que vimos anteriormente. Por ejemplo, si queremos seleccionar el primer *p* que aparece en el documento, podemos hacer:

```
var elems = $("p:eq(0)");
```

Esto busca todos los *p*s del documento y se queda con el primero de ellos. O si queremos todos *div*s que tengan la clase *cajita* y estén visibles, podemos usar:

```
var elems = $("div.cajita:visible");
```

Más selectores

Para una lista completa de todos los selectores que se pueden usar pueden ir a `http://docs.jquery.com/DOM/Traversing/Selectors#CSS_Selectors`.

Modificando las clases

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

- **addClass**: Nos permite agregarle una o más clases a un grupo de elementos, recibe como parámetro una cadena con las clases a agregar separadas por espacios.

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

- **addClass**: Nos permite agregarle una o más clases a un grupo de elementos, recibe como parámetro una cadena con las clases a agregar separadas por espacios.

```
$("div.link").addClass("active"); // Agregamos la clase
    active a todos los divs con clase link
$("p").addClass("bold big-font"); // Agregamos las
    clases bold y big-font a todos los ps del documento
```

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

- **addClass**: Nos permite agregarle una o más clases a un grupo de elementos, recibe como parámetro una cadena con las clases a agregar separadas por espacios.

```
$("div.link").addClass("active"); // Agregamos la clase
    active a todos los divs con clase link
$("p").addClass("bold big-font"); // Agregamos las
    clases bold y big-font a todos los ps del documento
```

- **removeClass**: Nos permite eliminar una o más clases de un elemento, recibe como parámetro las clases a eliminar.

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

- **addClass**: Nos permite agregarle una o más clases a un grupo de elementos, recibe como parámetro una cadena con las clases a agregar separadas por espacios.

```
$("div.link").addClass("active"); // Agregamos la clase
    active a todos los divs con clase link
$("p").addClass("bold big-font"); // Agregamos las
    clases bold y big-font a todos los ps del documento
```

- **removeClass**: Nos permite eliminar una o más clases de un elemento, recibe como parámetro las clases a eliminar.

```
$("div.active").removeClass("active open"); //
    Eliminamos las clases active y open
```

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

- **addClass**: Nos permite agregarle una o más clases a un grupo de elementos, recibe como parámetro una cadena con las clases a agregar separadas por espacios.

```
$("div.link").addClass("active"); // Agregamos la clase
    active a todos los divs con clase link
$("p").addClass("bold big-font"); // Agregamos las
    clases bold y big-font a todos los ps del documento
```

- **removeClass**: Nos permite eliminar una o más clases de un elemento, recibe como parámetro las clases a eliminar.

```
$("div.active").removeClass("active open"); //
    Eliminamos las clases active y open
```

- **hasClass**: Nos permite saber si un elemento tiene o no una clase. Recibe como parámetro la clase que se quiere verificar y devuelve **true** en caso que la contenga y **false** en caso contrario.

Modificando las clases

Para modificar las clases que posee un grupo de elementos JQuery nos proporciona los siguientes métodos:

- **addClass**: Nos permite agregarle una o más clases a un grupo de elementos, recibe como parámetro una cadena con las clases a agregar separadas por espacios.

```
$("div.link").addClass("active"); // Agregamos la clase
    active a todos los divs con clase link
$("p").addClass("bold big-font"); // Agregamos las
    clases bold y big-font a todos los ps del documento
```

- **removeClass**: Nos permite eliminar una o más clases de un elemento, recibe como parámetro las clases a eliminar.

```
$("div.active").removeClass("active open"); //
    Eliminamos las clases active y open
```

- **hasClass**: Nos permite saber si un elemento tiene o no una clase. Recibe como parámetro la clase que se quiere verificar y devuelve **true** en caso que la contenga y **false** en caso contrario.

```
$("div.link").hasClass("active"); // true
```

Modificando los atributos

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

```
var url = $("a.link").attr("href");
```

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

```
var url = $("a.link").attr("href");
```

attr también nos permite setear los atributos de uno o más elementos. Para esto disponemos de dos formas:

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

```
var url = $("a.link").attr("href");
```

  attr también nos permite setear los atributos de uno o más elementos. Para esto disponemos de dos formas:
  - Si queremos setear sólo un atributo, le podemos pasar como parámetros el nombre del atributo y el valor.

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

```
var url = $("a.link").attr("href");
```

attr también nos permite setear los atributos de uno o más elementos. Para esto disponemos de dos formas:

- Si queremos setear sólo un atributo, le podemos pasar como parámetros el nombre del atributo y el valor.

```
$("img").attr("src", "http://server.com/imagen.jpg")
    ;
```

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

```
var url = $("a.link").attr("href");
```

attr también nos permite setear los atributos de uno o más elementos. Para esto disponemos de dos formas:

- Si queremos setear sólo un atributo, le podemos pasar como parámetros el nombre del atributo y el valor.

```
$("img").attr("src", "http://server.com/imagen.jpg")
    ;
```

- Si en cambio queremos setear varios atributos al mismo tiempo le podemos pasar un objeto.

Modificando los atributos

Para modificar las atributos de los elementos disponemos de los siguientes métodos:

- **attr**: Nos permite obtener el valor de un atributo del primer elemento que coincida con el selector. Debemos pasarle como parámetro el nombre del atributo. Nos devuelve el valor del atributo si éste está seteado o undefined en caso de no estarlo.

```
var url = $("a.link").attr("href");
```

attr también nos permite setear los atributos de uno o más elementos. Para esto disponemos de dos formas:

- Si queremos setear sólo un atributo, le podemos pasar como parámetros el nombre del atributo y el valor.

```
$("img").attr("src", "http://server.com/imagen.jpg")
    ;
```

- Si en cambio queremos setear varios atributos al mismo tiempo le podemos pasar un objeto.

```
$("img#foto").attr({"alt" : "Una foto", "title" : "
    con title"});
```

Modificando los atributos - continuación

Modificando los atributos - continuación

- **removeAttr**: Nos permite eliminar una o más clases de un elemento, recibe como parámetro las clases a eliminar.

Modificando los atributos - continuación

- **removeAttr**: Nos permite eliminar una o más clases de un elemento, recibe como parámetro las clases a eliminar.

```
$("div.active").removeClass("active open"); //
    Eliminamos las clases active y open
```

Modificando el css

Modificando el css

Para modificar los estilos de un elemento siemplemente usamos el método **css** del objeto que hace referencia al elemento a modificar.

Modificando el css

Para modificar los estilos de un elemento siemplemente usamos el método **css** del objeto que hace referencia al elemento a modificar. Hay dos formas de llamar al método css, una es pasandole dos argumentos: la propiedad a modificar y el valor nuevo de dicha propiedad:

Modificando el css

Para modificar los estilos de un elemento siemplemente usamos el método **css** del objeto que hace referencia al elemento a modificar. Hay dos formas de llamar al método css, una es pasandole dos argumentos: la propiedad a modificar y el valor nuevo de dicha propiedad:

```
$("selector").css("propiedad", "valor");
```

Modificando el css

Para modificar los estilos de un elemento siemplemente usamos el método **css** del objeto que hace referencia al elemento a modificar. Hay dos formas de llamar al método css, una es pasandole dos argumentos: la propiedad a modificar y el valor nuevo de dicha propiedad:

```
$("selector").css("propiedad", "valor");
```

O, si queremos modificar varias propiedades a la vez, un objeto con las propiedades a modificar como claves y los nuevos valores de esas propiedades como los valores de dichas claves:

Modificando el css

Para modificar los estilos de un elemento siemplemente usamos el método **css** del objeto que hace referencia al elemento a modificar. Hay dos formas de llamar al método css, una es pasandole dos argumentos: la propiedad a modificar y el valor nuevo de dicha propiedad:

```
$("selector").css("propiedad", "valor");
```

O, si queremos modificar varias propiedades a la vez, un objeto con las propiedades a modificar como claves y los nuevos valores de esas propiedades como los valores de dichas claves:

```
$("#box").css({
    "height" : "300px",
    "width" : "400px"
});
```

Modificando el css

Para modificar los estilos de un elemento siemplemente usamos el método **css** del objeto que hace referencia al elemento a modificar. Hay dos formas de llamar al método css, una es pasandole dos argumentos: la propiedad a modificar y el valor nuevo de dicha propiedad:

```
$("selector").css("propiedad", "valor");
```

O, si queremos modificar varias propiedades a la vez, un objeto con las propiedades a modificar como claves y los nuevos valores de esas propiedades como los valores de dichas claves:

```
$("#box").css({
    "height" : "300px",
    "width" : "400px"
});
```

Algo que hay que tener en cuenta es que las propiedades que en css tienen nombres compuestos, separados por "-" en JQuery deben ir todas las palabras juntas, borrando los "-", y cada nueva palabra debe comenzar con una mayúscula. Por ejemplo: "margin-left" pasa a ser "marginLeft".

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

```
if(display == "none") {
    alert("Est  oculto");
} else {
    alert("Es visible");
}
```

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

```
if(display == "none") {
    alert("Est  oculto");
} else {
    alert("Es visible");
}
```

Ahora podríamos hacer que el elemento se muestre o se oculte dapendiendo de si está siendo mostrado o no.

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

```
if(display == "none") {
    alert("Est  oculto");
} else {
    alert("Es visible");
}
```

Ahora podríamos hacer que el elemento se muestre o se oculte dapendiendo de si está siendo mostrado o no.
Para mostrar un elemento podríamos usar:

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

```
if(display == "none") {
    alert("Est  oculto");
} else {
    alert("Es visible");
}
```

Ahora podríamos hacer que el elemento se muestre o se oculte dapendiendo de si está siendo mostrado o no.
Para mostrar un elemento podríamos usar:

```
$("#box").css("display", "block");
```

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

```
if(display == "none") {
    alert("Est  oculto");
} else {
    alert("Es visible");
}
```

Ahora podríamos hacer que el elemento se muestre o se oculte dapendiendo de si está siendo mostrado o no.
Para mostrar un elemento podríamos usar:

```
$("#box").css("display", "block");
```

Esto funcionaría, pero JQuery nos proporciona un método que hace esto de forma automática. Este método es **show** y se usa de la siguiente forma:

El método css tambiés nos devuelve el valor de una propiedad del estilo si le pasamos como parámetro una cadena con el nombre de dicha propiedad. Por ejemplo, si queremos saber si un elemento está siendo mostrado en la página podríamos hacer:

```
var display = $("#box").css("display");
```

```
if(display == "none") {
    alert("Est  oculto");
} else {
    alert("Es visible");
}
```

Ahora podríamos hacer que el elemento se muestre o se oculte dapendiendo de si está siendo mostrado o no.
Para mostrar un elemento podríamos usar:

```
$("#box").css("display", "block");
```

Esto funcionaría, pero JQuery nos proporciona un método que hace esto de forma automática. Este método es **show** y se usa de la siguiente forma:

```
$("#box").show();
```

Análogamente disponemos del método **hide**, que oculta un elemento del DOM:

Análogamente disponemos del método **hide**, que oculta un elemento del DOM:

```
$("#box").hide();
```

Análogamente disponemos del método **hide**, que oculta un elemento del DOM:

```
$("#box").hide();
```

Esto sería lo mismo que hacer:

Análogamente disponemos del método **hide**, que oculta un elemento del DOM:

```
$("#box").hide();
```

Esto sería lo mismo que hacer:

```
$("#box").css("display", "none");
```

Análogamente disponemos del método **hide**, que oculta un elemento del DOM:

```
$("#box").hide();
```

Esto sería lo mismo que hacer:

```
$("#box").css("display", "none");
```

Ahora uniendo todo, podríamos crear una función toggle a la que le pasamos una cadena con el selector de un elemento y oculta dicho elemento si este está siendo mostrado o lo muestra si no. Esta función quedaría como (en el siguiente slide):

```
function cambiar(selector) {
    var elem = $(selector);

    if(elem.css("display") == "none") {
        elem.show(); // Mostramos el elemento
    } else {
        elem.hide(); // Ocultamos el elemento
    }
}

cambiar("#box");
```

```
function cambiar(selector) {
    var elem = $(selector);

    if(elem.css("display") == "none") {
        elem.show(); // Mostramos el elemento
    } else {
        elem.hide(); // Ocultamos el elemento
    }
}

cambiar("#box");
```

Esta función que hicimos ya existe en JQuery y se llama toggle, se usa de la siguiente forma:

```
function cambiar(selector) {
    var elem = $(selector);

    if(elem.css("display") == "none") {
        elem.show(); // Mostramos el elemento
    } else {
        elem.hide(); // Ocultamos el elemento
    }
}

cambiar("#box");
```

Esta función que hicimos ya existe en JQuery y se llama toggle, se usa de la siguiente forma:

```
$("#box").toggle();
```

Eventos

Eventos

Los eventos son “avisos” que son lanzados cuando ocurre algún tipo de cambio sobre los elementos del DOM.

Eventos

Los eventos son "avisos" que son lanzados cuando ocurre algún tipo de cambio sobre los elementos del DOM.
Existen varios tipos de eventos. Algunos responden a acciones que realiza el usuarios, como: "click", "hover", "focus", "submit", etc; mientras que otros son consecuencia de cambios en la página, como: "ready", "load".

Eventos

Los eventos son "avisos" que son lanzados cuando ocurre algún tipo de cambio sobre los elementos del DOM.
Existen varios tipos de eventos. Algunos responden a acciones que realiza el usuarios, como: "click", "hover", "focus", "submit", etc; mientras que otros son consecuencia de cambios en la página, como: "ready", "load".
Muchas veces nos interesa realizar una acción cuando cierto evento ocurre. Para esto JQuery nos proporciona las siguientes funciones:

Eventos

Los eventos son "avisos" que son lanzados cuando ocurre algún tipo de cambio sobre los elementos del DOM.
Existen varios tipos de eventos. Algunos responden a acciones que realiza el usuarios, como: "click", "hover", "focus", "submit", etc; mientras que otros son consecuencia de cambios en la página, como: "ready", "load".
Muchas veces nos interesa realizar una acción cuando cierto evento ocurre. Para esto JQuery nos proporciona las siguientes funciones:

- **bind**: recibe dos parámentros, una cadena con el evento que se quiere "escuchar" y una función que se va a ejecutar cada vez que el evento es lanzado. Por ejemplo si queremos mostrar un alert cada vez que se hace click en algún *a* de la página podemo hacer:

```
$("a").bind("click", function(event) {alert("Se ha
    presionado un a");});
```

- **unbind**: Produce que una función que se enlazó a un evento usando **bind** deje de ejecutarse cada vez que se lanza dicho evento. Esto sólo funcionara si la función que se enlazó al evento no era una función anonima.

- **unbind**: Produce que una función que se enlazó a un evento usando **bind** deje de ejecutarse cada vez que se lanza dicho evento. Esto sólo funcionara si la función que se enlazó al evento no era una función anonima.

```
function alertarClick(event) {alert("Se ha presionado un
    link");}

$("a").bind("click", alertarClick);
$("a").unbind("click", alertarClick);
```

- **unbind**: Produce que una función que se enlazó a un evento usando **bind** deje de ejecutarse cada vez que se lanza dicho evento. Esto sólo funcionara si la función que se enlazó al evento no era una función anonima.

```
function alertarClick(event) {alert("Se ha presionado un
    link");}

$("a").bind("click", alertarClick);
$("a").unbind("click", alertarClick);
```

Si en cambio se quiere eliminar todas las funciones que están enlazadas a cualquier evento de un grupo de elementos usamos:

- **unbind**: Produce que una función que se enlazó a un evento usando **bind** deje de ejecutarse cada vez que se lanza dicho evento. Esto sólo funcionara si la función que se enlazó al evento no era una función anonima.

```
function alertarClick(event) {alert("Se ha presionado un
    link");}

$("a").bind("click", alertarClick);
$("a").unbind("click", alertarClick);
```

Si en cambio se quiere eliminar todas las funciones que están enlazadas a cualquier evento de un grupo de elementos usamos:

```
$("a").unbind();
```

- **unbind**: Produce que una función que se enlazó a un evento usando **bind** deje de ejecutarse cada vez que se lanza dicho evento. Esto sólo funcionara si la función que se enlazó al evento no era una función anonima.

```
function alertarClick(event) {alert("Se ha presionado un
    link");}

$("a").bind("click", alertarClick);
$("a").unbind("click", alertarClick);
```

Si en cambio se quiere eliminar todas las funciones que están enlazadas a cualquier evento de un grupo de elementos usamos:

```
$("a").unbind();
```

O si se quiere eliminar todas las funciones enlazadas a un evento especifico de un grupo de elementos podemos usar:

- **unbind**: Produce que una función que se enlazó a un evento usando **bind** deje de ejecutarse cada vez que se lanza dicho evento. Esto sólo funcionara si la función que se enlazó al evento no era una función anonima.

```
function alertarClick(event) {alert("Se ha presionado un
    link");}

$("a").bind("click", alertarClick);
$("a").unbind("click", alertarClick);
```

Si en cambio se quiere eliminar todas las funciones que están enlazadas a cualquier evento de un grupo de elementos usamos:

```
$("a").unbind();
```

O si se quiere eliminar todas las funciones enlazadas a un evento especifico de un grupo de elementos podemos usar:

```
$("a").unbind("click");
```

Existen varios atajos para bindear eventos, algunos son:

Existen varios atajos para bindear eventos, algunos son:

- **click**:

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

- **submit**:

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

- **submit**:

```
$("form").bind("submit", function(event){});
$("form").submit(function(event){});
```

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

- **submit**:

```
$("form").bind("submit", function(event){});
$("form").submit(function(event){});
```

- **change**:

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

- **submit**:

```
$("form").bind("submit", function(event){});
$("form").submit(function(event){});
```

- **change**:

```
$("select[name=pais]").bind("change", function(event){})
    ;
$("select[name=pais]").change(function(event){});
```

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

- **submit**:

```
$("form").bind("submit", function(event){});
$("form").submit(function(event){});
```

- **change**:

```
$("select[name=pais]").bind("change", function(event){})
    ;
$("select[name=pais]").change(function(event){});
```

- **ready**:

Existen varios atajos para bindear eventos, algunos son:

- **click**:

```
$("a").bind("click", function(event){});
$("a").click(function(event){});
```

- **submit**:

```
$("form").bind("submit", function(event){});
$("form").submit(function(event){});
```

- **change**:

```
$("select[name=pais]").bind("change", function(event){})
    ;
$("select[name=pais]").change(function(event){});
```

- **ready**:

```
$(document).bind("ready", function(event){});
$(document).ready(function(event){});
```

- **hover**: es un caso especial en donde se le pueden pasar dos funciones como parámetros en vez de sólo una. La primera función se va a ejecutar cuando el puntero del mouse entre en el elemento y la segunda cuando éste salga.

- **hover**: es un caso especial en donde se le pueden pasar dos funciones como parámetros en vez de sólo una. La primera función se va a ejecutar cuando el puntero del mouse entre en el elemento y la segunda cuando éste salga.

```
$("a").bind("mouseenter", function(e){alert("entro");});
$("a").bind("mouseleave", function(e){alert("salio");});

$("a").hover(
    function(e){alert("entro");},
    function(e){alert("salio");}
);
```

- **hover**: es un caso especial en donde se le pueden pasar dos funciones como parámetros en vez de sólo una. La primera función se va a ejecutar cuando el puntero del mouse entre en el elemento y la segunda cuando éste salga.

```
$("a").bind("mouseenter", function(e){alert("entro");});
$("a").bind("mouseleave", function(e){alert("salio");});

$("a").hover(
    function(e){alert("entro");},
    function(e){alert("salio");}
);
```

El objeto Event

Las funciones que enlazamos a los eventos reciben como primer parámetro un objeto que representa al evento lanzado. Este objeto posee, entre otras cosas, una referencia al elemento que lanzó el objeto y algunas funciones que nos pueden ser de utilidad. Dentro de ellas, *preventDefault* nos permite evitar que el objeto realice la acción que tiene asociada para ese evento por defecto.

Fin =).