Tema 2. Tecnologías del cliente. jQuery

Programación web

Boni García

Curso 2017/2018

Índice

Índice

Introducción

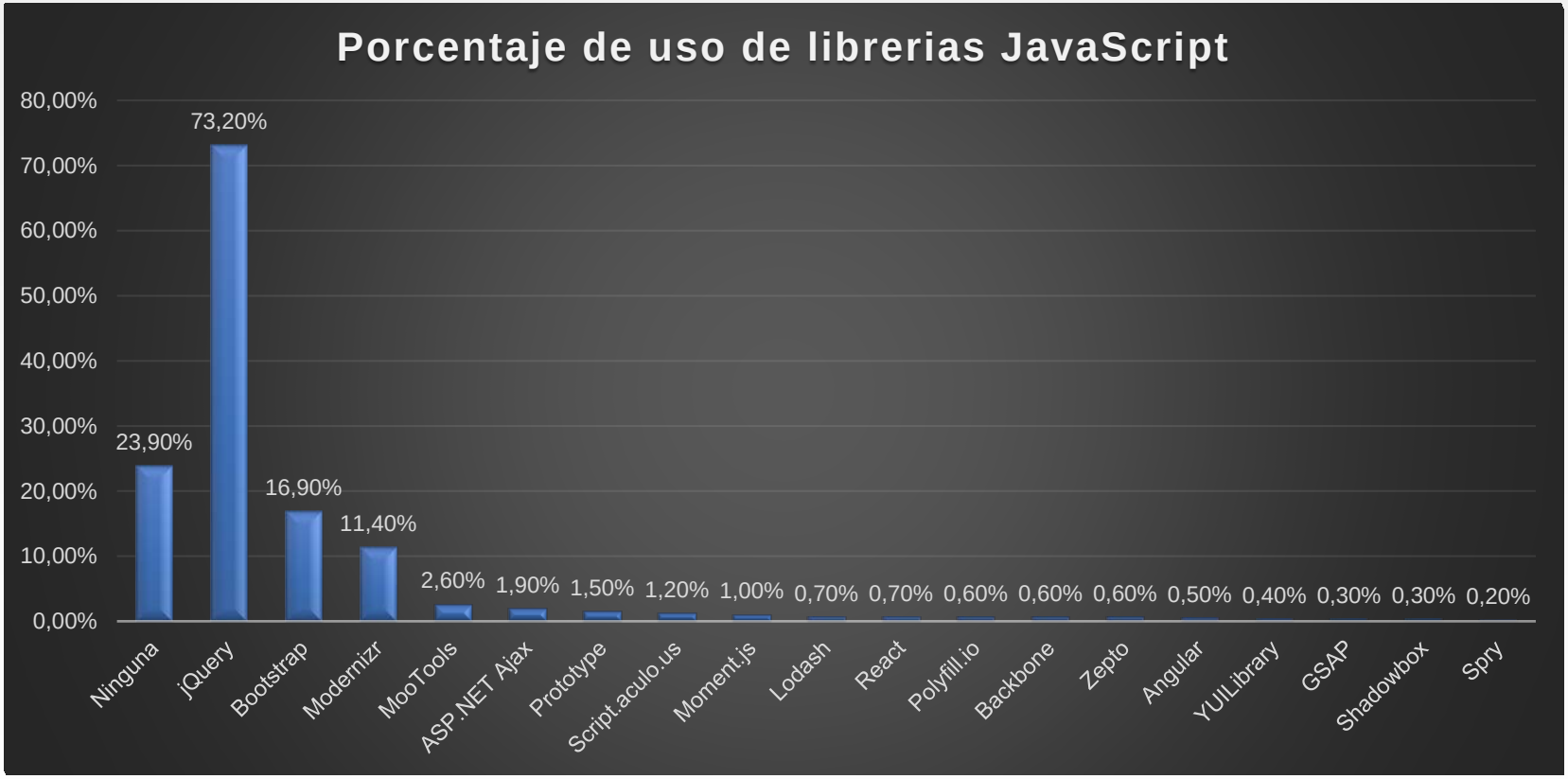
- **jQuery** es una librería **JavaScript** cuyo objetivo es simplificar y unificar el código JavaScript en los diferentes navegadores
  - Unifica las diferencias que existen en los navegadores
  - Tiene una API fluida (*fluent API*) para reducir el uso de variables locales y poder escribir scripts en una línea
  - Existen funcionalidades adicionales en forma de plugins, de forma que la librería se mantiene pequeña

http://jquery.com

Introducción

- jQuery permite:
  - Manipulación del DOM
  - Manipulación de CSS
  - Manipulación de eventos HTML
  - Efectos y animaciones
  - AJAX
  - Utilidades varias
- jQuery es la librería JavaScript más popular

Introducción



http://w3techs.com/technologies/overview/javascript_library/all
(12/03/2018)

Introducción

Versiones

- Hay tres versiones diferentes de jQuery: 1, 2, y 3
- La versión 1 está reservada para versiones antiguas de navegadores:
  - Internet Explorer 6, 7, y 8
  - Opera 12.1x
  - Safari 5.1+
- La versión 2 da soporte a navegadores modernos:
  - Chrome, Firefox, Edge, Safari: última versión estable y versión justo anterior
  - Internet Explorer: 9+
  - Opera: versión estable
  - Navegadores móviles: Safari en iOS 7+, Android browser 4.0+

Introducción

Versiones

- Diferencias entre jQuery 1 y 2:
  - Funcionalmente son idénticas
  - La versión 2 están más optimizadas que la 1
    - Reducción de tamaño de las librerías
    - Aumento de rendimiento
- La versión 3 incorpora nuevas características (*features*) y sustituye a la versión 2.x, esto es, da soporte optimizado para navegadores modernos
- Si queremos dar soporte a navegadores antiguos, habrá que usar jQuery 1.x

https://jquery.com/browser-support/

http://jquery.com/download/

Introducción

¿Cómo usamos jQuery?

- Versión estable (marzo de 2018): **3.3.1**
- Podemos usar jQuery enlazando con una red de distribución de contenidos (CDN, *Content Delivery Network*):
  - jQuery CDN: https://code.jquery.com/

```
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
```

  - Google CDN: https://developers.google.com/speed/libraries/devguide

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
```

En la versiones *minificadas* se eliminan bytes innecesarios (espacios, sangrías) para que el tamaño de la librería sea menor y se reduzca así el tiempo de carga

Sintaxis

- Con jQuery se seleccionan elementos HTML (*query*) y ser realizan acciones sobre estos elementos. La sintaxis básica general es:

```
$(selector).action();
```

… donde:

- `$` : Operador genérico para jQuery
- `(selector)` : Elemento(s) sobre los que realización la acción
- `action()` : Acción a ser ejecutada en los elementos seleccionados

- Ejemplos:

```
$("p").hide(); // hides all <p> elements
$(".test").hide(); //hides all elements with class="test"
$("#test").hide(); // hides the element with id="test"
```

Sintaxis

Selectores jQuery

- Los selectores jQuery están basados en CSS
- Algunos ejemplos típicos:

```
$("*") // Selects all elements
$("p.intro") // Selects all <p> elements with class="intro"
$("p:first") // Selects the first <p> element
$("ul li:first") // Selects the first <li> element of the first <ul>
$("ul li:first-child") // Selects the first <li> element of every <ul>
$("[href]") //Selects all elements with an href attribute
$("a[target='_blank']") //Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']") // Selects all <a> elements with a target attribute value NOT to "_blank"
$(":button") // Selects all <button> elements and <input> elements of type="button"
$("tr:even") // Selects all even <tr> elements
$("tr:odd") // Selects all odd <tr> elements
```

Sintaxis

Eventos

- La mayoría de eventos HTML (`click`, `keypress`, `submit`, ...) tienen su método jQuery equivalente. Por ejemplo:
  - Capturar el evento click en cualquier párrafo de la página:

```
$("p").click(function() {
   // action goes here!!
});
```

  - Ejecutar función cuando se completa la carga del documento:

```
$(document).ready(function() {
   // jQuery methods go here...
});
```

```
$(function() {
   // jQuery methods go here...
});
```

Nueva sintaxis introducida en jQuery 3

Efectos

Mostrar y ocultar

- Funciones `hide`, `show`, `toogle`. Ocultan/muestran elementos realizando una animación respecto al tamaño y opacidad elemento:

```
$(selector).hide(speed, callback);
$(selector).show(speed, callback);
$(selector).toggle(speed, callback);
```

  - `hide`: para ocultar elemento(s)
  - `show`: para mostrar elemento(s)
  - `toogle`: para conmutar la visibilidad de elemento(s) (si está oculto se muestra, si visible se ocula)
  - `speed`: (opcional) velocidad a la que se muestra/oculta elemento(s). Puede ser: `"slow"`, `"fast"`, o el valor numérico en milisegundos de la velocidad
  - `callback`: (opcional) función llamada al concluir el efecto de mostrar u ocultar

Efectos

Mostrar y ocultar

- Funciones `fadeIn`, `fadeOut`, `fadeToggle`, `fadeTo`. Ocultan y muestran elementos realizando una animación respecto solamente a la transparencia del elemento en cuestión:

```
$(selector).fadeIn(speed,callback);
$(selector).fadeOut(speed,callback);
$(selector).fadeToggle(speed,callback);
$(selector).fadeTo(speed,opacity,callback);
```

  - `fadeIn`: mostrar elemento(s) que permanecían invisibles
  - `fadeOut`: mostrar elemento(s) que permanecían visibles
  - `fadeToggle`: conmutar la visibilidad de un elemento(s) que permanecían invisibles
  - `fadeTo`: establece visibilidad final (`opacity` es un valor numérico entre 0 y 1 que define la transparencia, donde 1 = solido y 0 = invisible)
  - `speed` y `callback`: mismo funcionamiento que en `hide`, `show`, `toogle`

Efectos

Mostrar y ocultar

- Funciones `slideDown`, `slideUp`, `slideToggle`. Ocultan y muestran elementos realizan un efecto desplegable:

```
$(selector).slideDown(speed,callback);
$(selector).slideUp(speed,callback);
$(selector).slideToggle(speed,callback);
```

  - `slideDown`: mostrar elemento(s) que permanecían invisibles
  - `slideUp`: ocultar elemento(s) que permanecían visibles
  - `slideToggle`: conmutar la visibilidad de un elemento(s)
  - `speed` y `callback`: mismo funcionamiento que en `hide`, `show`, `toogle`

Efectos

Mostrar y ocultar

- Ejemplo:


```
<!DOCTYPE html>
<html>
<head>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script>
   $(function() {
      $("#toggle").click(function() {
         $("#p1").toggle("slow");
      });
      $("#fade").click(function() {
         $("#p2").fadeToggle("slow");
      });
      $("#slide").click(function() {
         $("#p3").slideToggle("slow");
      });
   });
</script>
<style>
div {
   text-align: center;
   background-color: orange;
   border: solid 1px black;
   padding: 50px;
   display: none;
}
</style>
</head>
<body>
<button id="toggle">Toggle</button>
<div id="p1">Hello world!</div>
<button id="fade">Fade</button>
<div id="p2">Hello world!</div>
<button id="slide">Slide</button>
<div id="p3">Hello world!</div>
</body>
</html>
```

Efectos

Animaciones

- La función `animate` permite manipular propiedades CSS realizando animaciones varias:

```
$(selector).animate({params},speed,callback);
```

- Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script>
   $(function() {
      $("button").click(function() {
         var div = $("div");
         div.animate({
            left : '200px'
         }, "slow");
         div.animate({
            fontSize : '3em',
            opacity : '0.2'
         }, 3000);
       });
   });
</script>
</head>
<body>
<button>Start Animation</button>
<div style="background: lime; position: absolute;">HELLO</div>
</body>
</html>
```

Manipulación del DOM

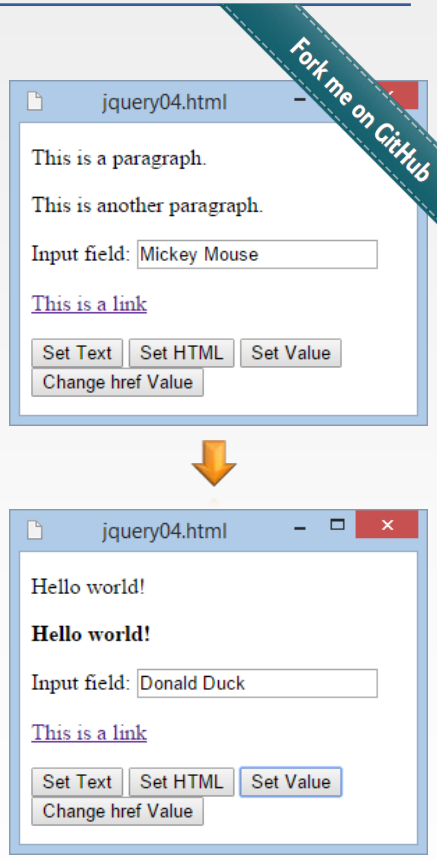
Lectura/escritura

- jQuery proporciona una serie de método para manipular el DOM de forma sencilla y universal (se ocupa de las posibles diferencias entre diferentes navegadores)
- Métodos de lectura/escritura:
  - `text()`: lee o escribe el contenido textual de un elemento
  - `html()`: lee o escribe el contenido HTML de un elemento
  - `val()`: lee o escribe el valor de un elemento de formulario
  - `attr()`: lee o escribe el valor de un atributo de un elemento

Manipulación del DOM

- Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script>
$(function() {
   $("#btn1").click(function() {
      $("#test1").text("Hello world!");
   });
   $("#btn2").click(function() {
      $("#test2").html("<b>Hello world!</b>");
   });
   $("#btn3").click(function() {
      $("#test3").val("Donald Duck");
   });
   $("#btn4").click(function(){
       $("#test4").attr("href", "https://www.u-tad.com/");
   });
});
</script>
</head>
<body>
<p id="test1">This is a paragraph.</p>
<p id="test2">This is another paragraph.</p>
<p>Input field: <input type="text" id="test3" value="Mickey Mouse"></p>
<p><a href="http://jquery.com/" id="test4">This is a link</a></p>

<button id="btn1">Set Text</button>
<button id="btn2">Set HTML</button>
<button id="btn3">Set Value</button>
<button id="btn4">Change href Value</button>
</body>
</html>
```



Manipulación del DOM

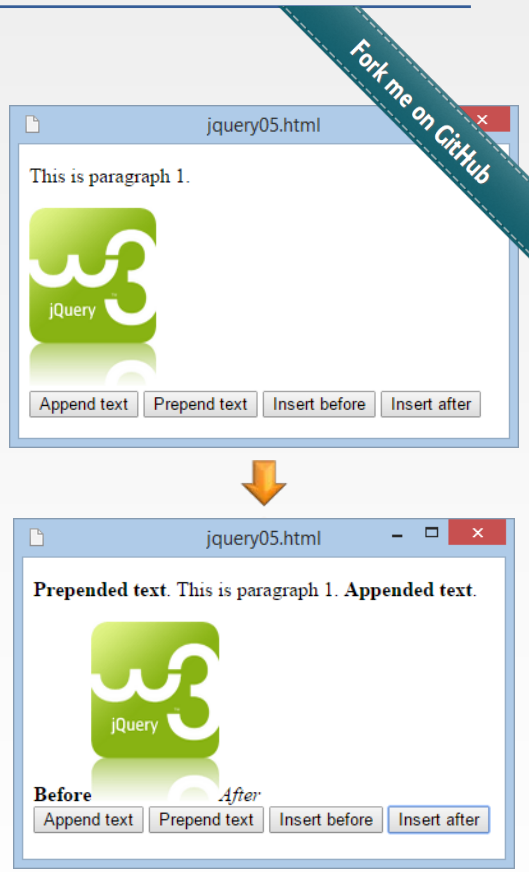
Añadir/eliminar

- Métodos para añadir o eliminar nuevos elementos:
  - `append()`: inserta HTML al final de un elemento
  - `prepend()`: inserta HTML al principio de un elemento
  - `after()`: inserta HTML antes de un elemento
  - `before()`: inserta HTML después de un elemento
  - `remove()`: elimina un elemento HTML y todos sus hijos
  - `empty()`: elimina un todos los hijos de un elemento HTML

Manipulación del DOM

- Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script>
$(function() {
   $("#btn1").click(function() {
      $("p").append(" <b>Appended text</b>.");
   });
   $("#btn2").click(function() {
      $("p").prepend("<b>Prepended text</b>. ");
   });
   $("#btn3").click(function() {
      $("img").before("<b>Before</b>");
   });
   $("#btn4").click(function() {
      $("img").after("<i>After</i>");
   });
});
</script>
</head>
<body>
<p>This is paragraph 1.</p>
<img src="w3jquery.gif" width="100„ height="140"><br>
<button id="btn1">Append text</button>
<button id="btn2">Prepend text</button>
<button id="btn3">Insert before</button>
<button id="btn4">Insert after</button>
</body>
</html>
```



Manipulación del DOM

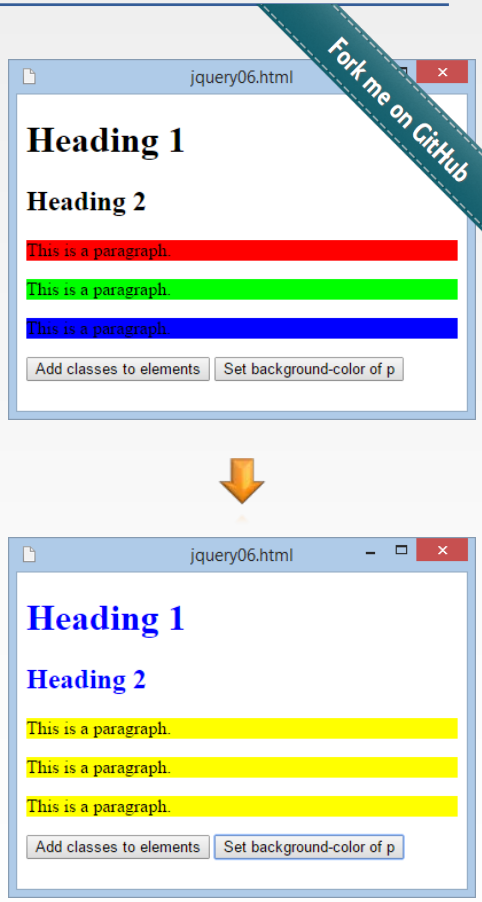
Manipulación de CSS

- Métodos para manipular CSS:
  - `addClass()`: añade una o más clases CSS a un elemento
  - `removeClass()`: elimina una o más clases CSS de un elemento
  - `toggleClass()`: conmuta una o más clases CSS de un elemento
  - `css()`: lee o modifica una o más propiedades CSS de un elemento

Manipulación del DOM

- Ejemplo:

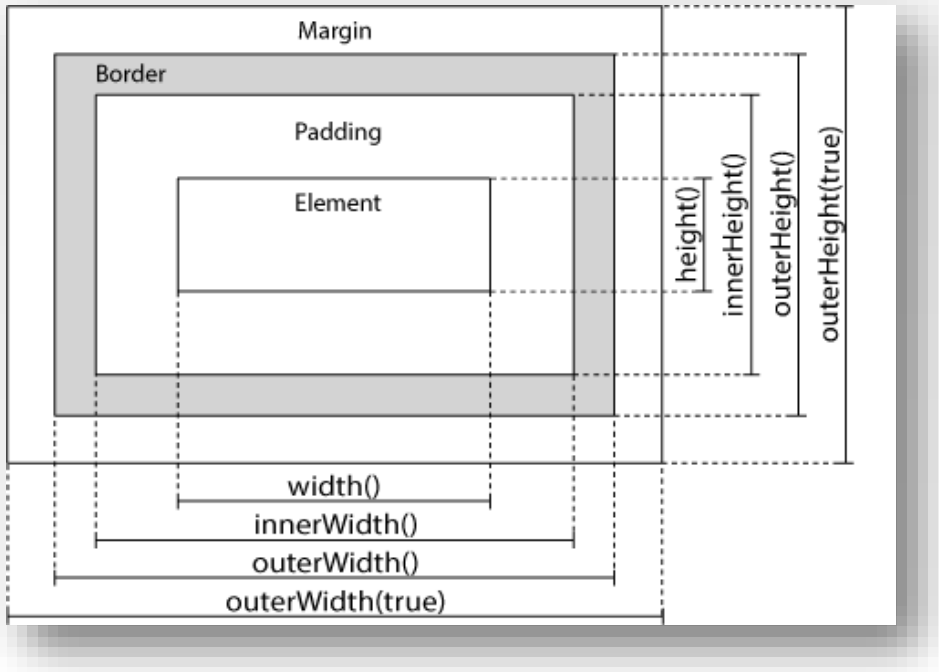
```
<!DOCTYPE html>
<html>
<head>
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script>
$(function() {
   $("#btn1").click(function() {
      $("h1, h2").addClass("blue");
   });

   $("#btn2").click(function() {
      $("p").css("background-color", "yellow");
   });
});
</script>
<style>
.blue {
   color: blue;
}
</style>
</head>
<body>
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<p style="background-color: #ff0000">This is a paragraph.</p>
<p style="background-color: #00ff00">This is a paragraph.</p>
<p style="background-color: #0000ff">This is a paragraph.</p>
<button id="btn1">Add classes to elements</button>
<button id="btn2">Set background-color of p</button>
</body>
</html>
```



Manipulación del DOM

Modelo de cajas

- Métodos para manipular el método de cajas CSS:
  - `width()`
  - `height()`
  - `innerWidth()`
  - `innerHeight()`
  - `outerWidth()`
  - `outerHeight()`



Manipulación del DOM

Búsquedas

- Métodos para realizar búsquedas de elementos padre:
  - `parent()`: devuelve el padre directo de un elemento
  - `parents()`: devuelve todos los elementos padre (hasta `<html>`)
  - `parentsUntil()`: devuelve todos los elementos padre (hasta un determinado elemento)
- Métodos para realizar búsquedas de elementos hijo:
  - `children()`: devuelve todos los hijos de un elemento
  - `find()`: devuelve todos los hijos de un elemento de un determinado tipo

Manipulación del DOM

Búsquedas

- Métodos para realizar búsquedas de elementos del mismo nivel:
  - `siblings()`: devuelve todos los elementos “hermanos”
  - `next()`: devuelve el elemento “hermano” siguiente
  - `nextAll()`: devuelve todos los elementos “hermano” siguientes
  - `nextUntil()`: devuelve todos los elementos “hermanos” siguientes hasta un determinado elemento
  - `prev()`: devuelve el elemento “hermano” anterior
  - `prevAll()`: devuelve todos los elementos “hermano” anteriores
  - `prevUntil()`: devuelve todos los elementos “hermanos” anteriores hasta un determinado elemento

Manipulación del DOM

Búsquedas

- Métodos para realizar filtrados en base a su posición dentro de un grupo:
  - `first()`: elige el primer elemento de un grupo
  - `last()`: elige el último elemento de un grupo
  - `eq()`: elige un elemento de un grupo dada su posición (0=primer lugar)
- Métodos para realizar filtrados en base a un criterio:
  - `filter()`: especifica una condición de filtrado (aplica el mismo criterio que para cualquier selector)
  - `not()`: lo contrario a `filter()`

AJAX

- Como sabemos, AJAX se proporciona de forma nativa en los navegadores modernos mediante el objeto `XMLHttpRequest`
- En navegadores antiguos como IE5 e IE6 todavía podríamos hacer uso de AJAX de la siguiente forma:

```
var xhr;
if (window.XMLHttpRequest) {
   xhr = new XMLHttpRequest();
} else {
    // code for IE5, IE6
   xhr = new ActiveXObject("Microsoft.XMLHTTP");
}
```

- jQuery proporciona una API unificada para hacer peticiones AJAX al servidor de manera sencilla

AJAX

- El método `ajax()` permite realizar peticiones AJAX a un servidor.

  Sintaxis: `$.ajax({name:value, name:value, ... });`

  …donde:

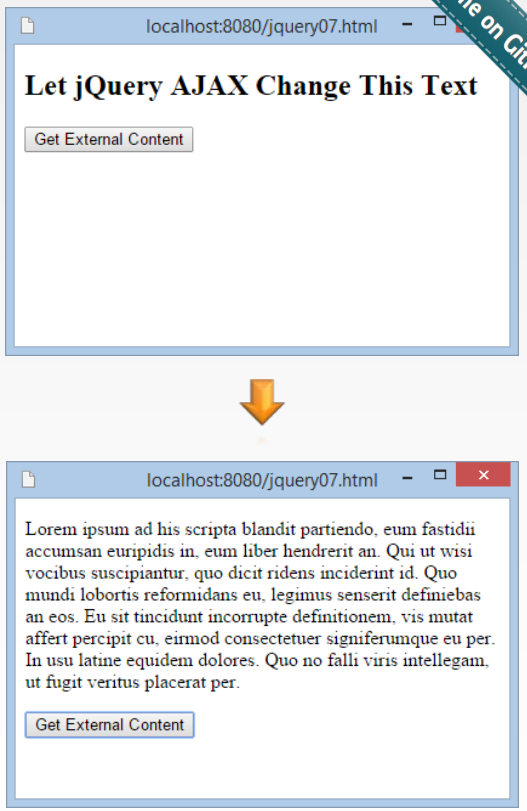
Name	*Value*
`url`	Recurso del servidor a la que va dirigida la petición AJAX
`success`	Función que se ejecuta cuando se recibe una respuesta correcta por parte del servidor
`error`	Función que se ejecuta cuando se recibe una respuesta errónea por parte del servidor
`data`	Datos a ser mandados hacia el servidor
`method`	Tipo de petición: `"GET"` (por defecto) o `"POST"`

http://api.jquery.com/jquery.ajax/

AJAX

- Ejemplo:

```
<!DOCTYPE html>
<html>

<head>
    <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        $(function () {
            $("button").click(function () {
                $.ajax({
                    url: "lore.html",
                    success: function (result) {
                        $("#div1").html(result);
                    }
                });
            });
        });
    </script>
</head>

<body>
    <div id="div1">
        <h2>Let jQuery AJAX Change This Text</h2> </div>
    <button>Get External Content</button>
</body>

</html>
```



AJAX

- El método `load()` permite cargar contenido de un servidor en segundo plano. Sintaxis:

```
$(selector).load(URL, data, callback);
```

- Ejemplo:

```
<!DOCTYPE html>
<html>

<head>
    <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        $(function () {
            $("button").click(function () {
                $("#div1").load("lore.html");
            });
        });
    </script>
</head>

<body>
    <div id="div1">
        <h2>Let jQuery AJAX Change This Text</h2>
    </div>
    <button>Get External Content</button>
</body>

</html>
```

AJAX

- Esta funcionalidad se puede usar para evitar la duplicación del *layout* en un sitio web estático:

```
<!DOCTYPE html>
<html>

<head>
    <link rel="stylesheet" href="site.css">
    <script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
    <script>
        $(function () {
            $("body").load("site-layout.html", function () {
                $(".main").load("site-content-page1.html");
            });
        });
    </script>
</head>

<body>
</body>

</html>
```

```
<div class="container">
    <div class="header">
        <h1>Header</h1>
    </div>
    <div class="content">
        <div class="nav">
            <a href="site-page1.html">Section 1</a><br>
            <a href="site-page2.html">Section 2</a><br>
        </div>
        <div class="main"></div>
    </div>
    <div class="footer">Copyright &copy; Company.com</div>
</div>
```

Cookies

- Liberia JavaScript para uso de Cookies (anteriormente basada en jQuery)

```
<!DOCTYPE html>
<html>

<head>
<script src="https://cdn.jsdelivr.net/npm/js-cookie@2/src/js.cookie.min.js"></script>
<script>
   // Write cookie
   Cookies.set("my_cookie", "new_value");

   // Read cookie
   var cookie = Cookies.get("my_cookie");
   console.log(cookie);

   // Delete cookie
   Cookies.remove("my_cookie");
   cookie = Cookies.get("my_cookie");
   console.log(cookie);
</script>
</head>

<body>
<h1>Using cookies...</h1>
</body>

</html>
```

https://github.com/js-cookie/js-cookie