



Universitat d'Alacant
Universidad de Alicante

XHTML DINAMICO AVANZADO (AJAX Y DOM)





AJAX

- Librerías estándares
- Ejemplos prácticos.





Bibliografía

- Ajax in Practice
- Visual Quickstart Guide CSS, DHTML, and Ajax, Fourth Edition
- DHTML Utopia. Modern Web Design Using JavaScript & DOM

Ajax: Un Nuevo acercamiento a las Aplicaciones Web

<http://www.ajaxhispano.com/ajax-nuevo-acercamiento-aplicaciones-web.html>

El objeto XMLHttpRequest

<http://www.programacionweb.net/articulos/articulo/?num=386>

Wikipedia

<http://es.wikipedia.org/wiki/Portada>

Librerías estándares - Prototype

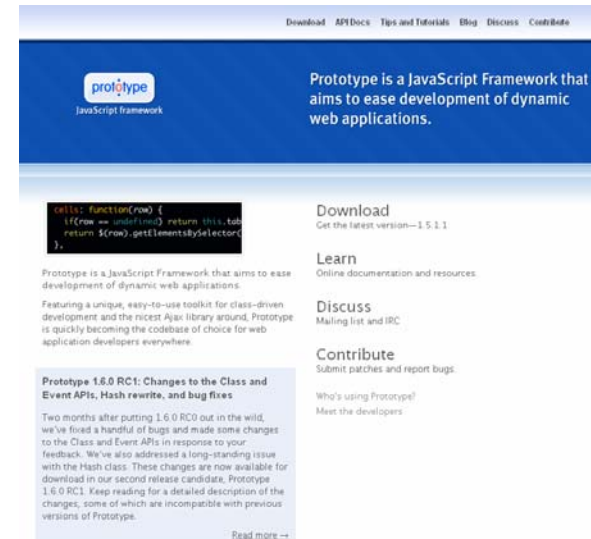
<http://www.prototypejs.org/>

Prototype es un framework desarrollado en JavaScript por Sam Stephenson para el desarrollo sencillo y dinámico de páginas Web.

Prototype nos simplifica gran parte del trabajo cuando se pretende desarrollar páginas altamente interactivas.

Proyectos basados en Prototype

- Ruby on Rails
(<http://www.rubyonrails.com/>)
- script.aculo.us , Thomas Fuchs
(<http://script.aculo.us/>)
- Rico
- (<http://openrico.org/>)



The screenshot shows the homepage of the Prototype JavaScript Framework. At the top, there is a navigation menu with links for Download, API Docs, Tips and Tutorials, Blog, Discuss, and Contribute. The main header features the Prototype logo and the text: "Prototype is a JavaScript Framework that aims to ease development of dynamic web applications." Below this, there is a code snippet:

```
$(this).function(row) {  
  if(row == undefined) return this.tab;  
  return $(row).getElementsbySelector();  
}
```

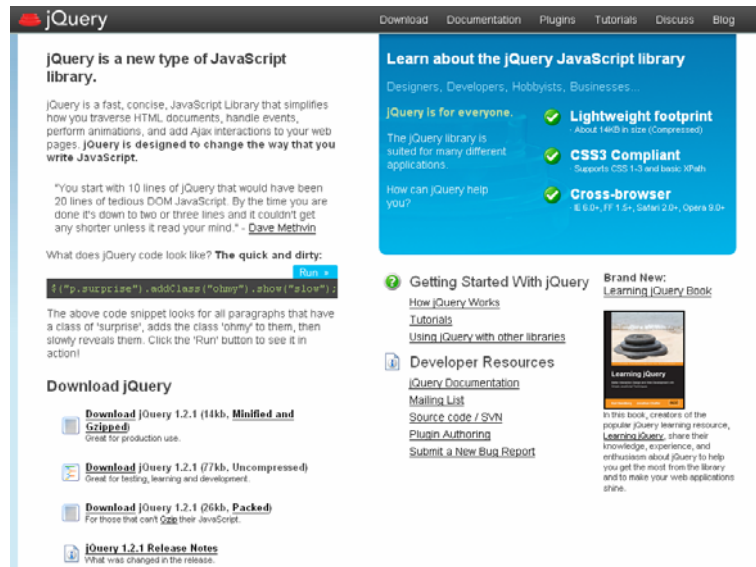
 The page is divided into several sections: "Download" (Get the latest version—1.5.1.1), "Learn" (Online documentation and resources), "Discuss" (Mailing list and IRC), and "Contribute" (Submit patches and report bugs). There is also a section for "Who's using Prototype?" and "Meet the developers". A "Read more" link is visible at the bottom of the page.

Librerías estándares - jQuery

<http://jquery.com/>

Es una liviana librería de JavaScript, pensada para interactuar con los elementos de una Web por medio del DOM.

La sencillez de su sintaxis y la poca extensión del código que necesitas escribir son las características más notables.

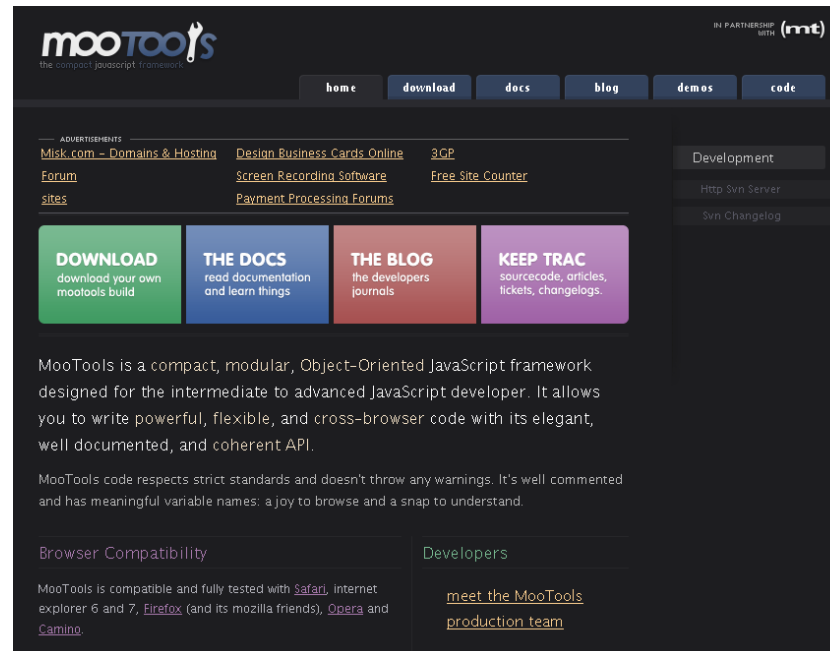


The screenshot shows the jQuery website homepage. The main heading is "jQuery is a new type of JavaScript library." Below this, there is a description of jQuery as a fast, concise JavaScript library that simplifies DOM traversal, event handling, and Ajax interactions. A quote from Dave Methwin is included: "You start with 10 lines of jQuery that would have been 20 lines of tedious DOM JavaScript. By the time you are done it's down to two or three lines and it couldn't get any shorter unless it read your mind." - Dave Methwin. There is a code snippet for a "quick and dirty" example: `$(function() { $(".surprise").addClass("show"); });` with a "Run" button. The "Download jQuery" section offers three options: "Download jQuery 1.2.1 (14kb, Minified and Gzipped)", "Download jQuery 1.2.1 (77kb, Uncompressed)", and "Download jQuery 1.2.1 (26kb, Packed)". There are also links for "jQuery 1.2.1 Release Notes" and "Getting Started With jQuery". A sidebar on the right highlights features like "Lightweight footprint", "CSS3 Compliant", and "Cross-browser".

Librerías estándares - Mootools

<http://mootools.net/>

Mootools, es una librería desarrollada en Javascript con la que la tarea de programar la parte funcional de una aplicación Web se convierte en una tarea más fácil, potente y cómoda.



The screenshot shows the Mootools website homepage. At the top left is the Mootools logo with the tagline "the compact javascript framework". To the right, it says "IN PARTNERSHIP WITH (mt)". Below the logo is a navigation menu with links for "home", "download", "docs", "blog", "demos", and "code". Underneath the navigation menu is an "ADVERTISEMENTS" section with several links. Below that is a row of four colored boxes: "DOWNLOAD" (green), "THE DOCS" (blue), "THE BLOG" (red), and "KEEP TRAC" (purple). Below these boxes is a paragraph of text describing Mootools as a compact, modular, Object-Oriented JavaScript framework. At the bottom, there are two columns: "Browser Compatibility" and "Developers".

mooTOOLS
the compact javascript framework

IN PARTNERSHIP WITH (mt)

home download docs blog demos code

ADVERTISEMENTS

Misk.com - Domains & Hosting Design Business Cards Online 3GP
Forum Screen Recording Software Free Site Counter
sites Payment Processing Forums

Development
Http Svn Server
Svn Changelog

DOWNLOAD
download your own mootools build

THE DOCS
read documentation and learn things

THE BLOG
the developers journals

KEEP TRAC
sourcecode, articles, tickets, changelogs.

MooTools is a compact, modular, Object-Oriented JavaScript framework designed for the intermediate to advanced JavaScript developer. It allows you to write powerful, flexible, and cross-browser code with its elegant, well documented, and coherent API.

MooTools code respects strict standards and doesn't throw any warnings. It's well commented and has meaningful variable names: a joy to browse and a snap to understand.

Browser Compatibility
MooTools is compatible and fully tested with Safari, Internet explorer 6 and 7, Firefox (and its mozilla friends), Opera and Camino.

Developers
[meet the MooTools production team](#)

Test de Velocidad entre Frameworks

<http://mootools.net/slickspeed/>



speed/validity selectors test for frameworks.

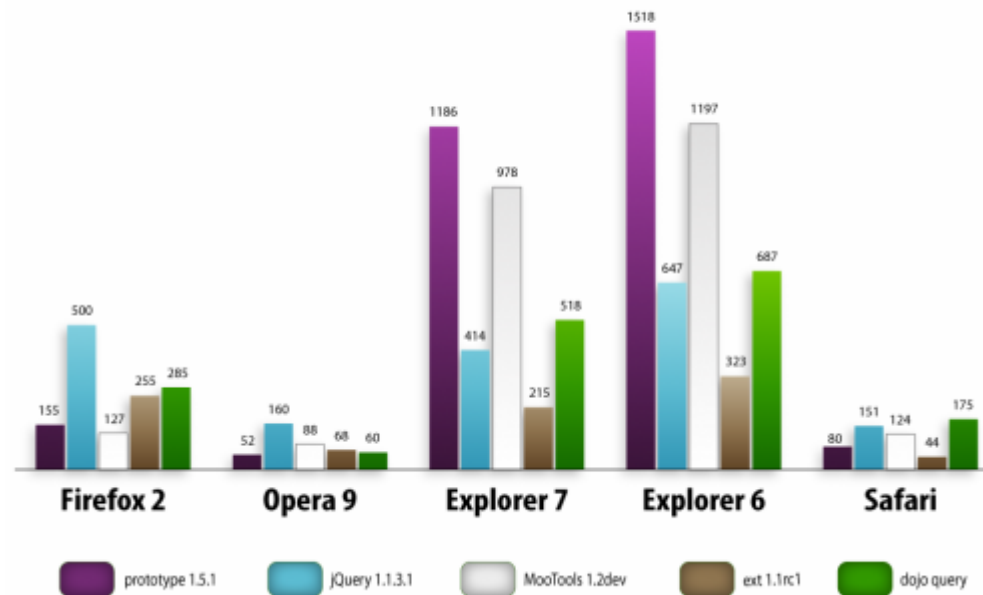
Every framework runs in his own iFrame, thus no conflicts can happen. Tests are run selector by selector, with an interval to prevent the browser from freezing. Tests are run in a neutral environment, no library or framework is included in the main javascript test, to avoid favoritism.

selectors	MooTools 993	Prototype 1.6rc0	jQuery 1.21
*			
div:only-child			
div:contains(CELIA)			
div:nth-child(even)			
div:nth-child(2n)			
div:nth-child(odd)			
div:nth-child(2n+1)			
div:nth-child(n)			
div:last-child			
div:first-child			
div > div			
div + div			
div ~ div			
body			
body div			
div			
div div			
div div div			
div, div, div			
div, a, span			

Test de Velocidad entre Frameworks

<http://www.yukei.net/2007/09/a-prueba-frameworks-javascript/>

Prueba de Selectores



Prueba realizada con SlickSpeed, <http://extjs.com/playpen/slickspeed/>

Mootools – Razones para elegirlo

Nos ofrece una serie de objetos con los que podremos trabajar más cómodamente. Además de esta serie de objetos, disponemos de facilidades para crear nuestros propios objetos y sobrecargarlos con las funcionalidades que MooTools nos ofrece.

- Es completamente modular y puedes personalizar lo que necesitas descargar para ahorrarte peso de javascript
- Leer el código de MooTools es como leer un libro, la versión con código incluido es realmente explícita y simple de entender.
- MooTools te permite desarrollar con un código orientado a objetos, esto debido a la capacidad que tiene de extender los objetos nativos del lenguaje.

Mootools – Descarga

<http://mootools.net/download>

Elementos necesarios para los ejercicios

- Core
- Element
- Element.Event y Element.Dimensions
- Window.DomReady
- Fx.Style
- Json.Remote

JSON

JSON, acrónimo de "[JavaScript Object Notation](#)", es un formato ligero para el intercambio de datos. **JSON** es un subconjunto de la notación literal de objetos de Javascript pero no requiere el uso de XML.

La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a [XML](#) en [AJAX](#). Una de las supuestas ventajas de **JSON** sobre [XML](#) como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador semántico de JSON.

En Javascript, JSON puede ser analizado trivialmente usando el procedimiento [eval\(\)](#), lo cual ha sido fundamental para la aceptación de JSON por parte de la comunidad de desarrolladores Ajax, debido a la ubicuidad de Javascript en casi cualquier navegador Web.

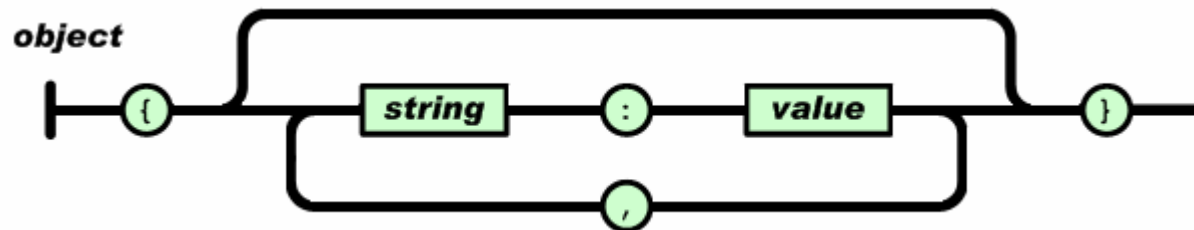
JSON

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un *objeto*, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

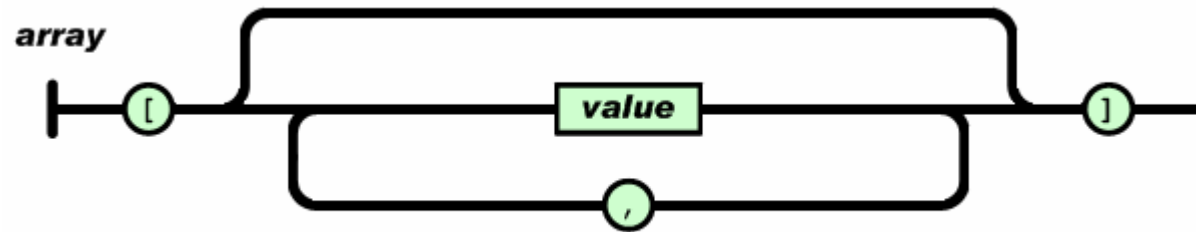
JSON

Un *objeto* es un conjunto desordenado de pares nombre/valor. Un objeto comienza con { (llave de apertura) y termine con } (llave de cierre). Cada nombre es seguido por : (dos puntos) y los pares nombre/valor están separados por , (coma).



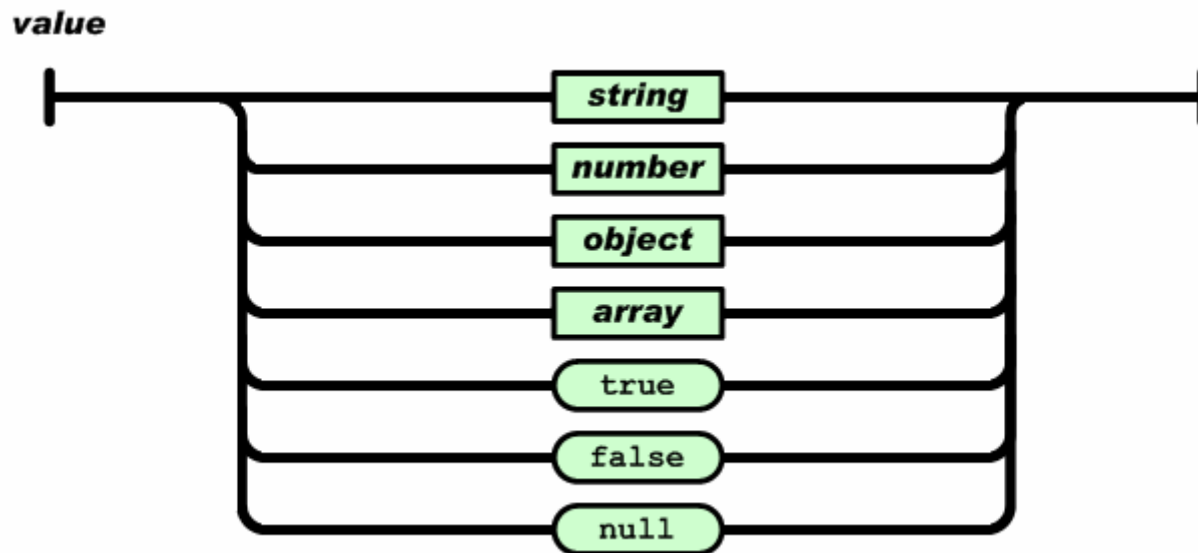
JSON

Un *array* es una colección de valores. Un array comienza con [(corchete izquierdo) y termina con] (corchete derecho). Los valores se separan por , (coma).



JSON

Un *valor* puede ser una *cadena de caracteres* con comillas dobles, o un *número*, o true o false o null, o un *objeto* o un *arreglo*. Estas estructuras pueden anidarse





JSON – Equivalencia con XML

```
{"menu": {  
  "id": "file",  
  "value": "File",  
  "popup": {  
    "menuitem": [  
      {"value": "New", "onclick": "CreateNewDoc()"},  
      {"value": "Open", "onclick": "OpenDoc()"},  
      {"value": "Close", "onclick": "CloseDoc()"}  
    ]  
  }  
}}
```

```
<menu id="file" value="File">  
  <popup>  
    <menuitem value="New" onclick="CreateNewDoc()" />  
    <menuitem value="Open" onclick="OpenDoc()" />  
    <menuitem value="Close" onclick="CloseDoc()" />  
  </popup>  
</menu>
```




JSON – Enlaces de Interés

- <http://es.wikipedia.org/wiki/JSON>
- <http://json.org/>
- <http://json.org/json-es.html>

Trabajar con JSON desde ASP

- <http://www.webdevbros.net/2007/04/26/generate-json-from-asp-datatypes/>
- <http://www.webdevbros.net/2007/08/21/json-utility-class-13-released/>



JSON – Solicitud AJAX

Formato

```
var jsonRequest = new Json.Remote("script que hace el servicio",  
    {onComplete: function(objeto json){  
        }}).send({variables a enviar});
```

Ejemplo

```
var jsonRequest = new Json.Remote("servicios/personas_texto.asp",  
    {onComplete: function(jsonpersonas){  
        acabaAccion('JSON', jsonpersonas.personas);  
        }}).send({'filtro': ''});
```

JSON – Formato datos

Formato

```
{“objeto”: [{“campo1”: valor1,“campo2”: “valor2”,...}]}
```

Ejemplo

```
{"personas": [{"idpersona": 3,"nombre": "ALFONSO","apellidos":  
"BENAVENT VICTORIA","email": "ABenavent@ua.es"}]}
```

MooTools – Funciones

each (Clase Array)

Recorre todos los elementos del array y como parámetro indicamos la función que podemos ejecutar para cada uno de los elementos del array.

Formato

```
array.each(function(item) {  
    alert(item.propiedad);  
});
```

Ejemplo

```
['apple','banana','lemon'].each(function(item, index) {  
    alert(index + " = " + item);  
});
```



JSON – Leer datos (Objetos)

Formato

```
ArrayObjetos.each(function(objeto) {  
  // Leemos las propiedades de cada objeto;  
});
```

Recorremos los datos

```
ppersonas.each(function(persona) {  
  alert(persona.nombre + " " + persona.apellidos + " (" +  
    persona.email + ")");  
});
```



JSON – Ejemplo 1

Crear un fichero html (ej1_json.html) que haga una llamada AJAX a un servicio JSON (servicios/personas_texto.asp) y mostremos con un alert el nombre, apellidos y correo del usuario.



JSON – Generar datos desde ASP

Descargar librería

<http://www.webdevbros.net/wp-content/uploads/2007/08/json13.zip>

Uso sencillo desde una consulta (RecordSet)

Formato

```
(new JSON).toJSON("identificación", RecordSet, false)
```

Ejemplo

```
Set oRS = Ocon.Execute(ssQL)
```

```
' Recorrer el cursor para mostrar los datos
```

```
  If (not oRS.EOF) then
```

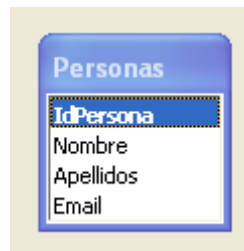
```
    response.write((new JSON).toJSON("personas", oRS, false))
```

```
  End If
```

JSON – Ejemplo 2

Crear un fichero html (ej2_json.html) que haga una llamada AJAX a un servicio JSON (servicios/personas.asp). Este ASP consulta los datos de la tabla Personas y devuelve todos los datos.

Cuando acabe la llamada, mostraremos un alert con el nombre, apellidos y correo, de cada uno de los alumnos.



Personas			
IdPersona			
Nombre			
Apellidos			
Email			

JSON – Generar datos desde ASP

Enviar nuestros propios datos con un diccionario

Formato

Crear un diccionario y añadir datos con `diccionario.add “nombre”, “valor”`

```
((new JSON).toJSON(" identificación ", arrayDiccionario, false))
```

Ejemplo

```
Dim dPer(1)
```

```
set dPer(0) = server.createObject("scripting.dictionary")
```

```
dPer(0).add "idpersona", "1"
```

```
set dPer(1) = server.createObject("scripting.dictionary")
```

```
dPer(1).add "idpersona", "2"
```

```
response.write((new JSON).toJSON("personas", dPer, false))
```

JSON – Ejemplo 3

Crear un fichero html (ej3_json.html) que haga una llamada AJAX a un servicio JSON (servicios/personas_3.asp).

Este ASP genera los datos de las personas con un array de diccionarios. Cuando acabe la llamada, mostraremos un alert con el nombre, apellidos y correo, de cada uno de los alumnos.

MooTools – Funciones

Función \$

Nos devuelve la referencia al objeto que estamos buscando, siempre y cuando este exista en la página. En caso de no existir devuelve false.

Formato

`$(‘mielemento’)`

Ejemplo

```
if ($(‘idTextarea’)) {  
    $(‘idTextarea’).value = “Hola”;  
}  
else {  
    alert(“No existe el textarea idTextarea”);  
}
```

MooTools – Funciones

Función \$\$

Nos devuelve una array de objetos que se ajustan a una etiqueta, #identificador, clase, etc que indiquemos

Formato

\$\$('etiqueta')

Ejemplo

\$\$('a') // Array con todas las etiquetas anchor de la página

\$\$('a.clase') // Array con todas las etiquetas anchor de la página
// que tengan como class, clase

MooTools – Funciones

Evento DomReady

Añadimos un evento especial a la ventana cuando el DOM está listo para poder trabajar con él.

Ejemplo

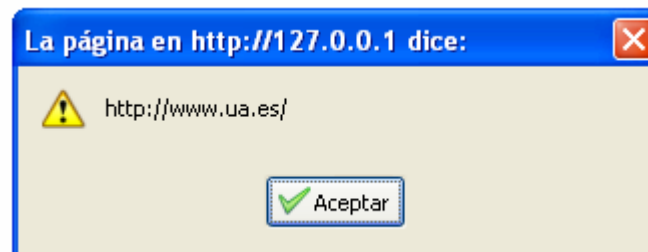
```
window.addEvent('domready',  
  function(){  
    alert('the dom is ready');  
  }  
);
```

Mootools – Ejemplo 4

Crear un fichero html (ej4_mootools.html) que contenga un listado de las Universidades de la Comunidad Valenciana (sacar datos de <http://www.ua.es/es/internet/listado.htm>) y que cuando esté listo el DOM se active un evento que muestre las urls de cada uno de los enlaces (propiedad href).

Listado de Universidades de la Comunidad Valenciana

- ♦ [Universitat d'Alacant / Universidad de Alicante](#)
- ♦ [Universitat Jaume I](#)
- ♦ [Universidad Miguel Hernández](#)
- ♦ [Universitat Politècnica de València](#)
- ♦ [Universitat de València](#)



MooTools – Funciones

injectBefore (Clase elemento)

Inserta un elemento antes del elemento actual.

Ejemplo

```
$(“idelemento”).injectBefore(“Elemento que va a ser ahora posterior”);
```



MooTools – Funciones

injectAfter (Clase elemento)

Inserta un elemento después del elemento actual.

Ejemplo

```
$(“idelemento”).injectAfter(“Elemento que va a ser ahora anterior”);
```


MooTools – Funciones

remove (Clase elemento)

Elimina el elemento actual.

Ejemplo

```
$(“idelemento”).remove();
```



MooTools – Funciones

setHTML (Clase elemento)

Indicamos el innerHTML del elemento.

Ejemplo

```
$(“idelemento”).setHTML(“Texto”);
```



MooTools – Funciones

getText (Clase elemento)

Obtenemos el inner text del elemento.

Ejemplo

```
alert($("#idelemento").getText());
```

MooTools – Funciones

new Element (Clase elemento)

Creamos un nuevo elemento. Debemos definir la etiqueta y los atributos que la forman.

Ejemplo

```
var el = new Element('div', {'style': 'color: red'});
```

Mootools – Ejemplo 5

Crear un fichero html (ej5_mootools.html) que contenga un listado de las Universidades de la Comunidad Valenciana (usar los mismos del ejercicio anterior) y que cuando esté listo el DOM se active un evento que cambie la posición de los dos primeros elementos de la lista y que reemplace los enlaces de los 3 últimos por etiquetas H1 en rojo

Listado de Universidades de la Comunidad Valenciana

- [Universitat Jaume I](#)
- [Universitat d'Alacant / Universidad de Alicante](#)
- **Universidad Miguel Hernández**
- **Universitat Politècnica de València**
- **Universitat de València**



MooTools – Funciones

injectInside (Clase elemento)

Inserta un elemento dentro del elemento actual.

Ejemplo

```
$(“idelemento”).injectInside(“Elemento en el que vamos a insertar”);
```

Mootools - Ejemplo 6

Crear un fichero html (ej6_mootools.html) que cuando esté listo el DOM se active un evento que cree una etiqueta div de 500px y fondo gris y que dentro de esta inserte 5 etiquetas div con cada una de las universidades.

Opcional: Añadir evento a las etiquetas para ir a la Web de la Universidad.

Universitat d'Alacant / Universidad de Alicante

Universitat Jaume I

Universidad Miguel Hernández

Universitat Politècnica de València

Universitat de València

JSON - Mootools - Ejemplo 7

Crear un fichero html (ej7_json_mootools.html) que disponga de una caja de texto que hará de filtro. Cuando el número de letras sea mayor de 3, hará una llamada AJAX a un servicio JSON (servicios/personas_7.asp) Este ASP consulta los datos de la tabla Personas dependiendo del filtro por nombre, apellidos o correo y devuelve todos los datos.

Cuando acabe la llamada, mostraremos una caja div (al estilo del ejercicio anterior) con los resultados que ha devuelto al servicio.