

Tema 3. Frameworks de desarrollo en el lado del cliente.

Desarrollo de Tecnologías Web:
Tecnologías en el Cliente (DTWTC)

*Master Universitario en Ingeniería de Servicios y
Aplicaciones Web*

Índice

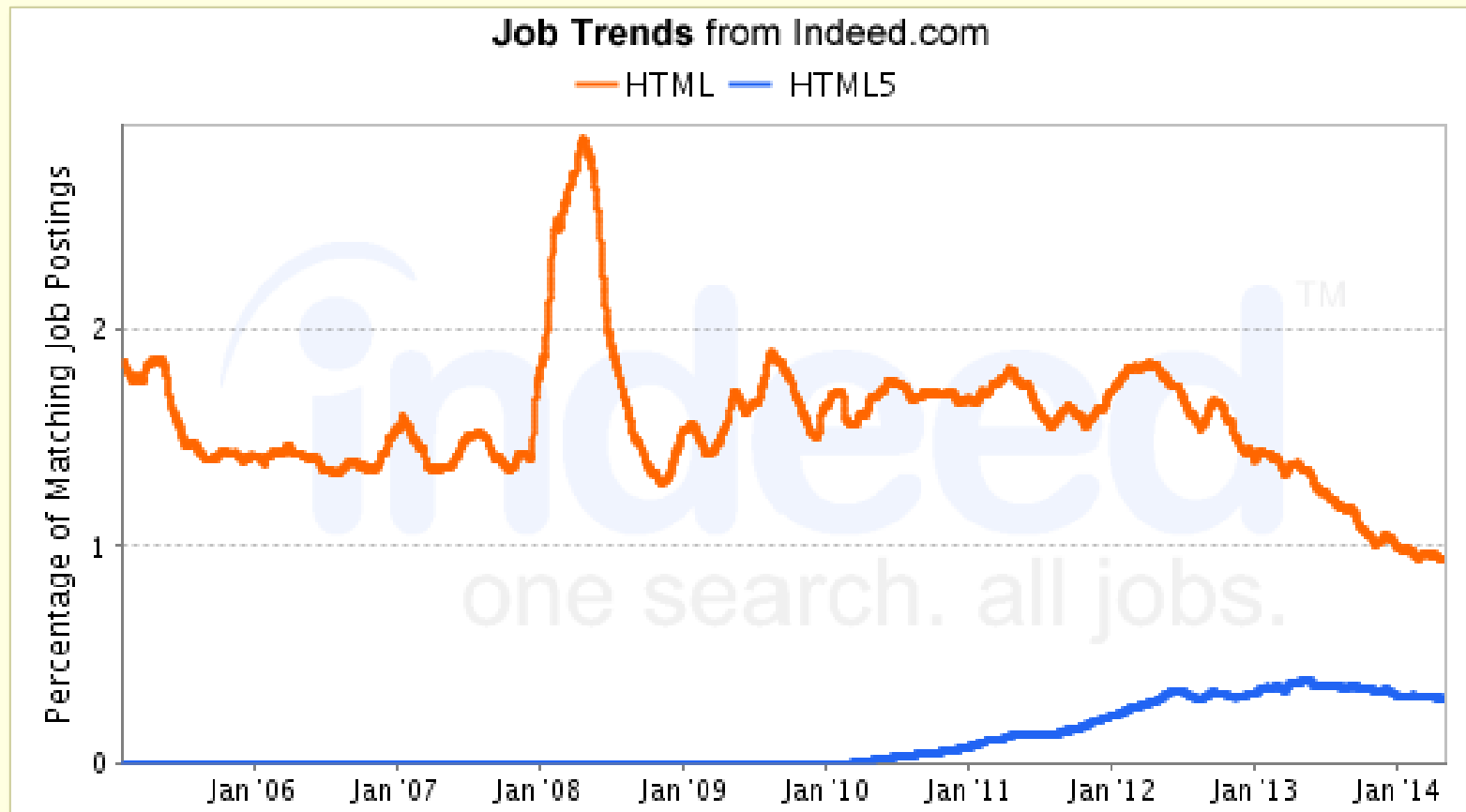
- Introducción
- JSON
- JQuery
- Otros Frameworks de JavaScript

Introducción

- Principales librerías para Javascript
 - JQuery
 - Ext JS
 - MooTools
 - YUI
 - Dojo
 - GWT
 - Node.js

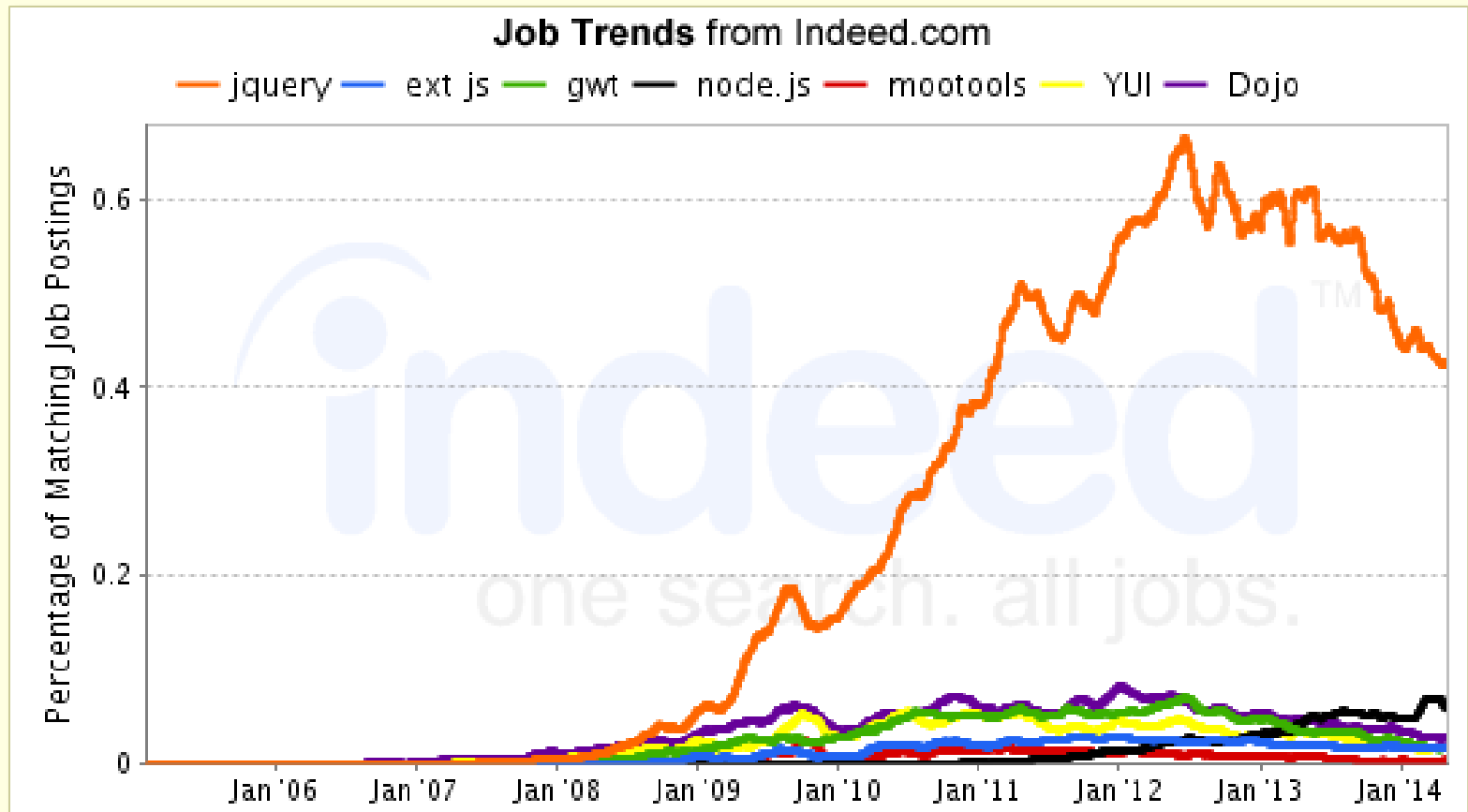
Job Trends from indeed.com

<http://www.indeed.com/jobtrends>



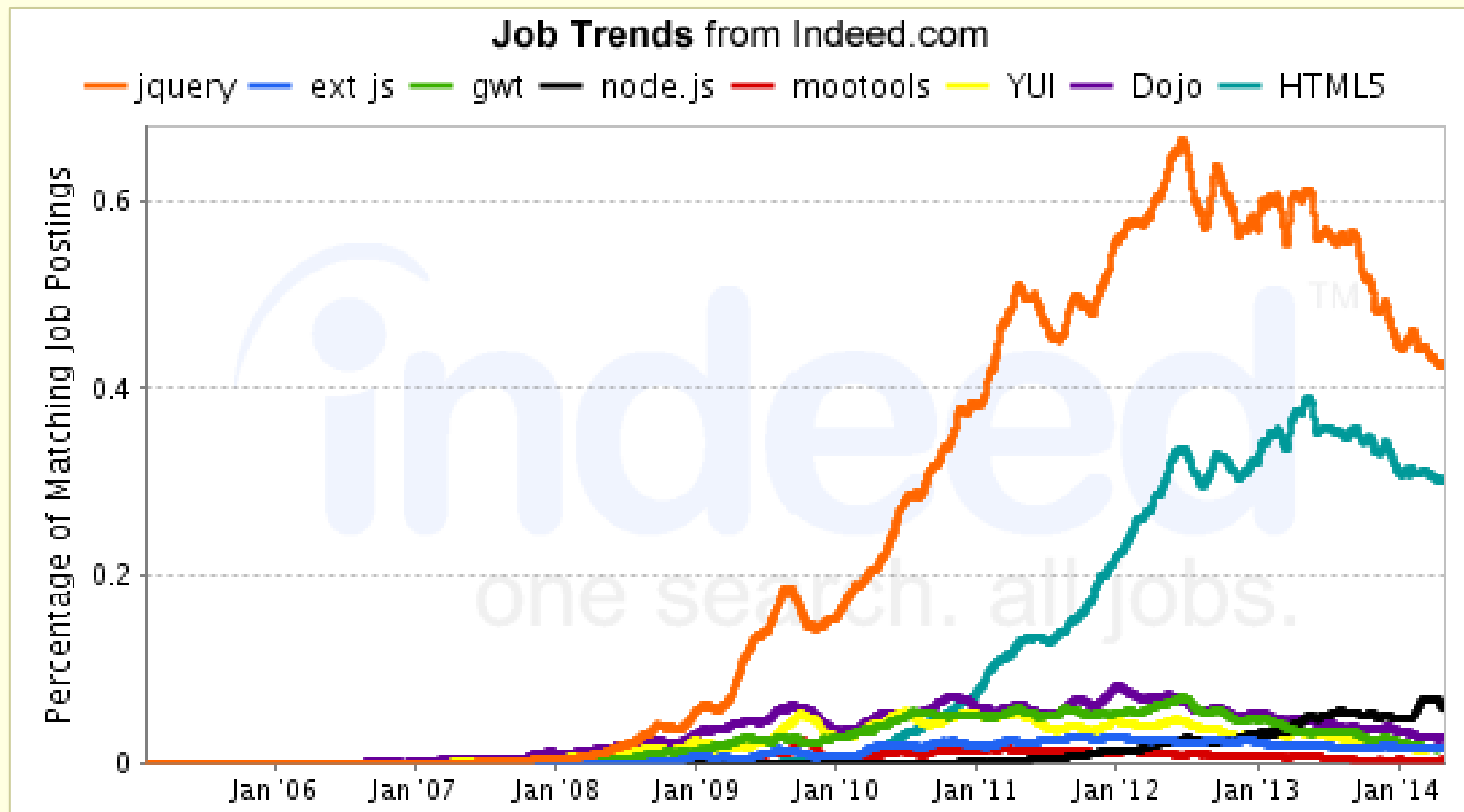
Job Trends from indeed.com

<http://www.indeed.com/jobtrends>



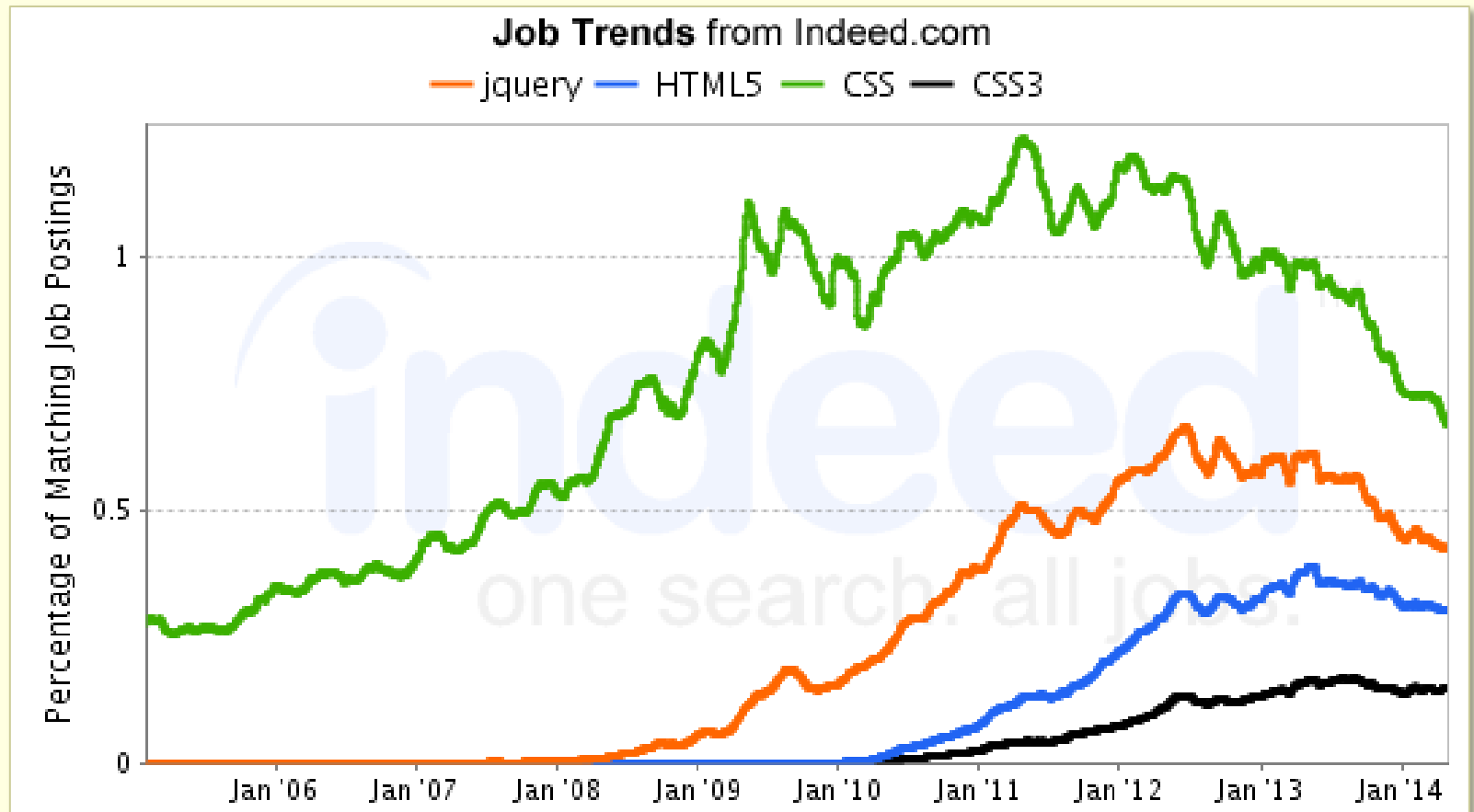
Job Trends from indeed.com

<http://www.indeed.com/jobtrends>



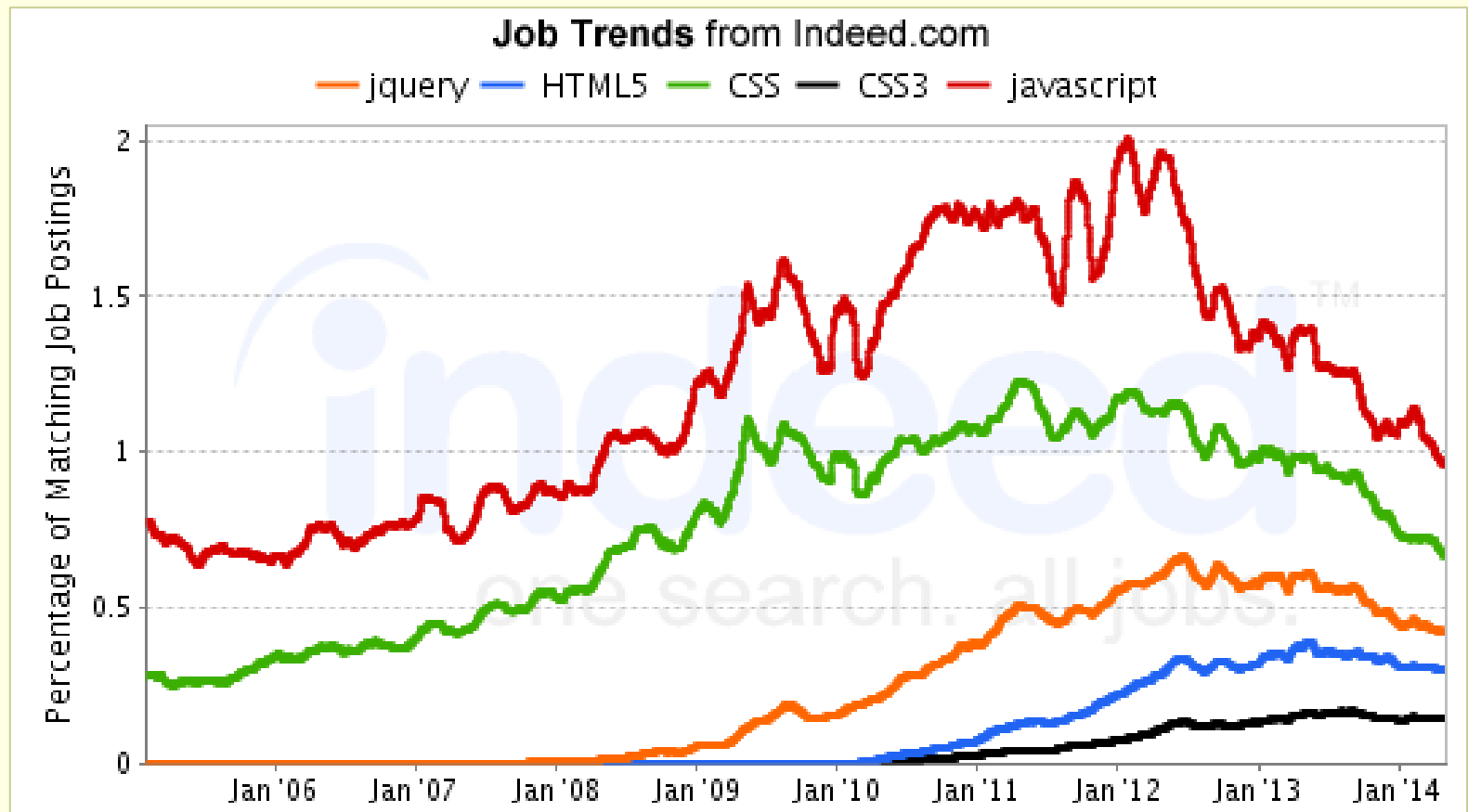
Job Trends from indeed.com

<http://www.indeed.com/jobtrends>



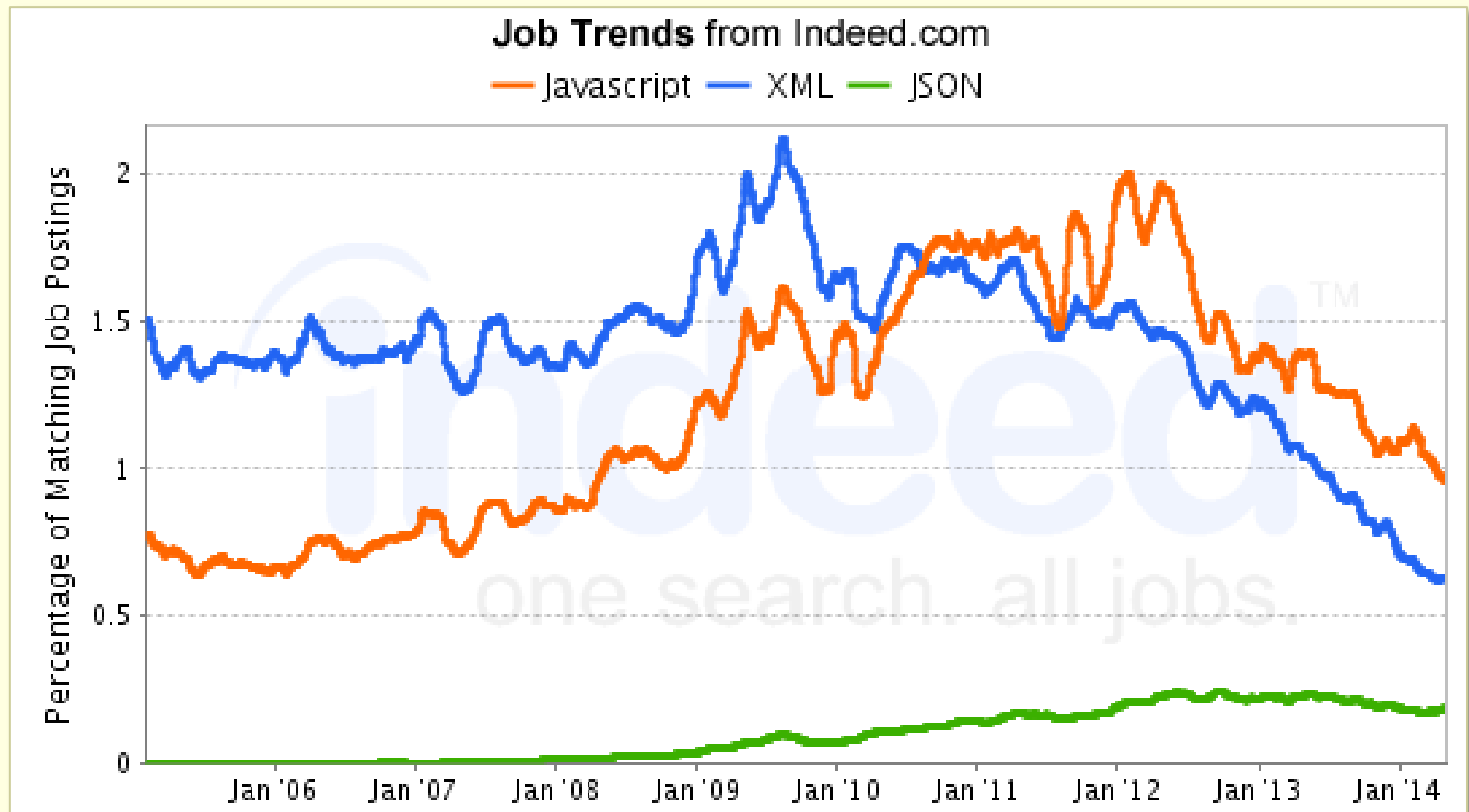
Job Trends from indeed.com

<http://www.indeed.com/jobtrends>



Job Trends from indeed.com

<http://www.indeed.com/jobtrends>



JSON

- ¿Qué es?
- Estructura
- Comparación frente a XML

JSON

- ¿Qué es JSON?
 - JavaScript Object Notation
 - Formato de intercambio de datos ligero (comparado con XML)
 - Basado en JavaScript (ECMA-262)
 - Posee un formato simple (para personas y para máquinas)
 - Formato de texto independiente del lenguaje de programación que lo utilice
 - Pensado para su uso en AJAX y facilitar las tareas de programación

JSON

- Está basado en dos estructuras:

- Objetos

- Colección de pares nombre/valor
 - Ejemplo:

```
{  
    "producto": "vino de mesa",  
    "cantidad": 12,  
    "precio": 4.95  
}
```

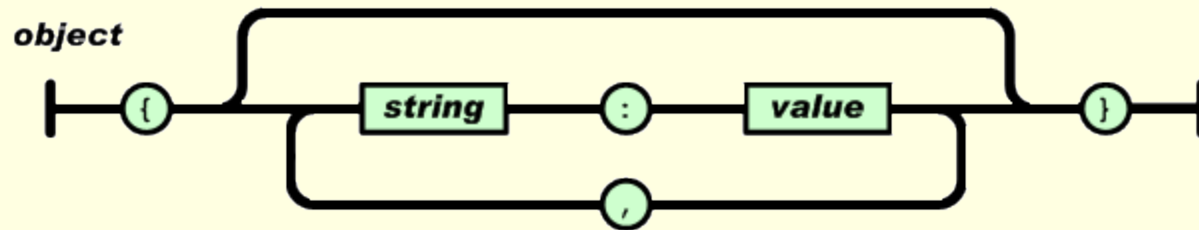
- Vectores (arrays)

- Lista ordenada de valores
 - Ejemplo:

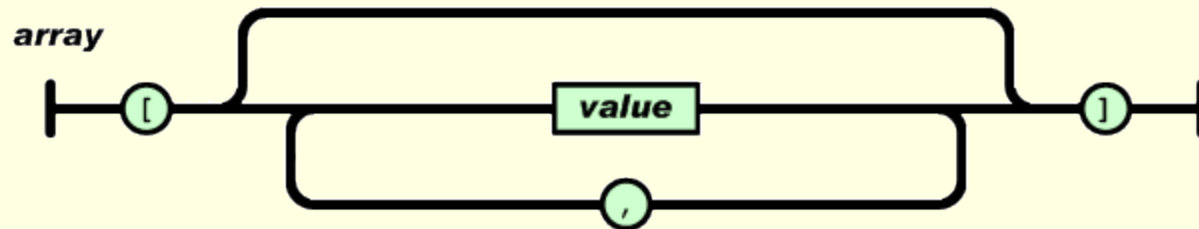
```
["vino de mesa", 12, 4.95]
```

JSON

■ Objetos

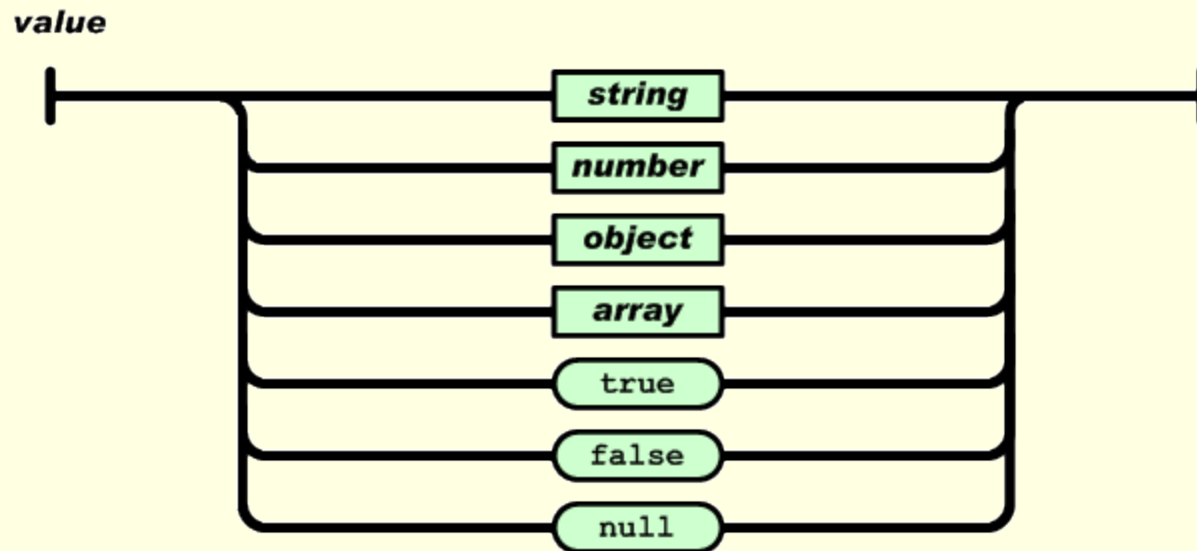


■ Arrays



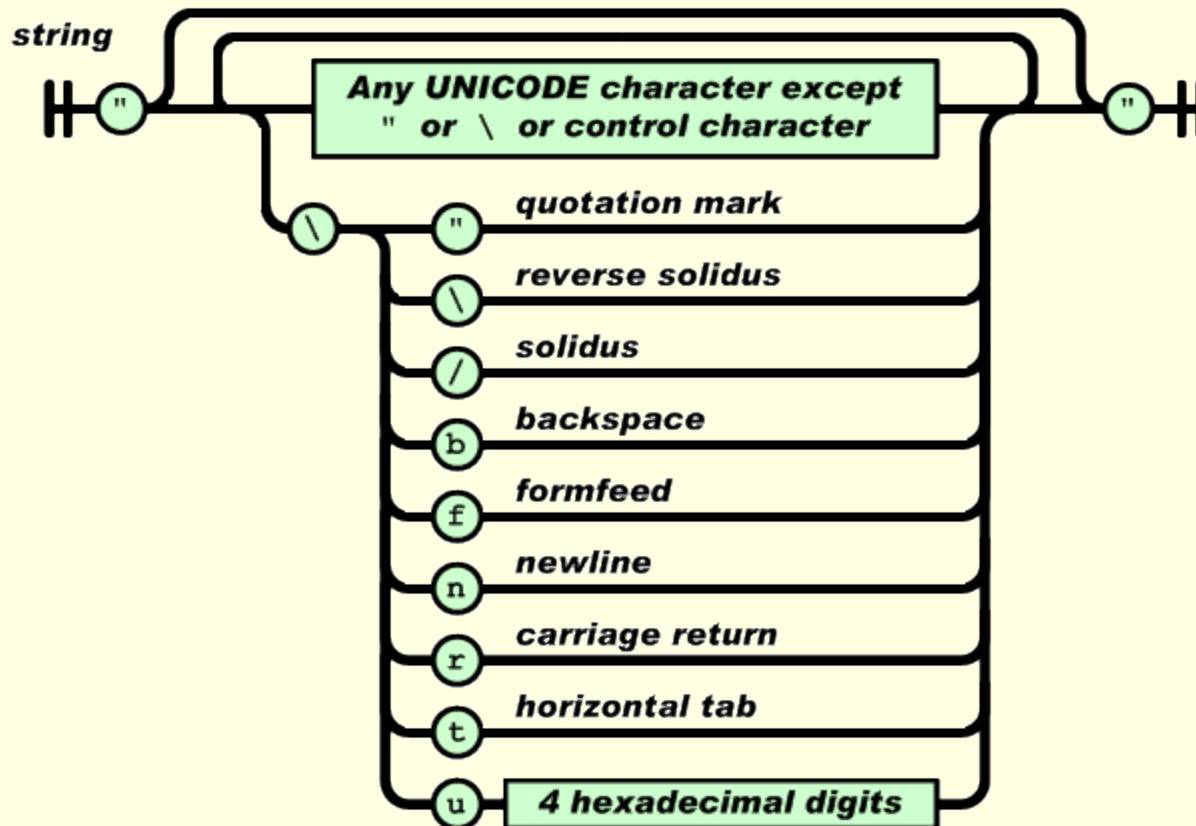
JSON

■ Valores



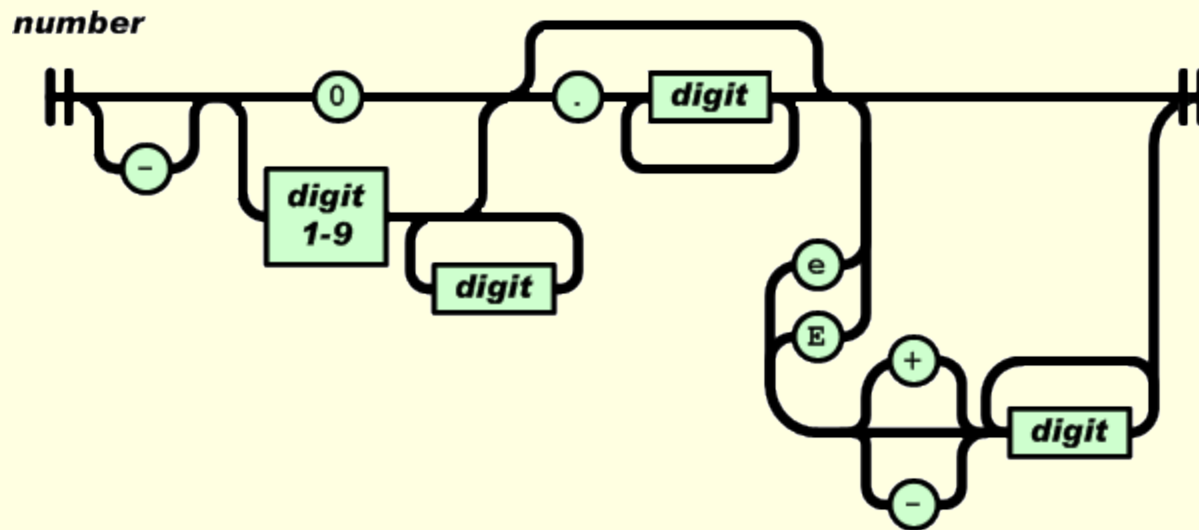
JSON

■ Cadenas



JSON

■ Números



JSON

■ Conversión de formatos

■ JSON → JavaScript

```
var myObject = eval('(' + myJSONtext + ')');  
var myObject = JSON.parse(myJSONtext);
```

■ JavaScript → JSON

```
var myJSONText = JSON.stringify(myObject);
```

JSON

- Es similar a XML en:
 - Ambos son auto-descriptivos, con valores que entienden humanos y máquinas
 - Ambos son jerárquicos
 - Ambos se pueden analizar (parsear) con muchos lenguajes de programación
 - Ambos se pueden utilizar para funciones AJAX (usando `httpXMLRequest`).

JSON

- Es diferente a XML en:
 - XML utiliza etiquetas con nombre al principio y al final, mientras que JSON sólo usa un nombre al principio del elemento
 - JSON es más rápido de escribir porque es menos estricto en cuanto a sintaxis
 - JSON se puede analizar (parsear) de forma sencilla con el procedimiento `eval()`
 - JSON permite el uso de vectores (arrays)
 - JSON no puede usar palabras reservadas para JavaScript

JSON

■ Comparación JSON – XML: XML

```
<?xml version='1.0' encoding='UTF-8'?>
<card>
  <fullname>Sean Kelly</fullname>
  <org>SK Consulting</org>
  <emailaddrs>
    <address type='work'>kelly@seankelly.biz</address>
    <address type='home' pref='1'>kelly@seankelly.tv</address>
  </emailaddrs>
  <telephones>
    <tel type='work' pref='1'>+1 214 555 1212</tel>
    <tel type='fax'>+1 214 555 1213</tel>
    <tel type='mobile'>+1 214 555 1214</tel>
  </telephones>
  <addresses>
    <address type='work' format='us'>1234 Main St Springfield,
      TX 78080-1216</address>
    <address type='home' format='us'>5678 Main St Springfield,
      TX 78080-1316</address>
  </addresses>
  <urls>
    <address type='work'>http://seankelly.biz/</address>
    <address type='home'>http://seankelly.tv/</address>
  </urls>
</card>
```

JSON

■ Comparación JSON – XML: **JSON**

```
{
  "fullname": "Sean Kelly",
  "org": "SK Consulting",
  "emailaddrs": [
    {"type": "work", "value": "kelly@seankelly.biz"},
    {"type": "home", "pref": 1, "value": "kelly@seankelly.tv"}
  ],
  "telephones": [
    {"type": "work", "pref": 1, "value": "+1 214 555 1212"},
    {"type": "fax", "value": "+1 214 555 1213"},
    {"type": "mobile", "value": "+1 214 555 1214"}
  ],
  "addresses": [
    {"type": "work", "format": "us",
      "value": "1234 Main StnSpringfield, TX 78080-1216"},
    {"type": "home", "format": "us",
      "value": "5678 Main StnSpringfield, TX 78080-1316"}
  ],
  "urls": [
    {"type": "work", "value": "http://seankelly.biz/"},
    {"type": "home", "value": "http://seankelly.tv/"}
  ]
}
```

JSON

■ Comparación evaluación JSON – XML

■ JSON

```
var my_object = {  
  "nombre": "Ramon Cirilo",  
  "empresa": "Universidad de Valencia"  
}
```

`my_object.nombre`

■ XML

```
<?xml version='1.0' encoding='UTF-8'?>  
<elemento>  
  <nombre>Ramon Cirilo</nombre>  
  <empresa>Universidad de Valencia</empresa>  
</elemento>
```

`document.getElementsByTagName('nombre')[0].firstchild`

JSON

■ Referencias

■ Introducción a JSON

- <http://www.json.org/>

■ JSON y Javascript

- <http://www.json.org/js.html>

■ Validador JSON

- <http://jsonlint.com/>
- <http://json.parser.online.fr/>
- <http://jsonviewer.stack.hu/>

■ Conversor XML-JSON

- <http://xmlgrid.net/jsonXml.html>
- <http://www.utilities-online.info/xmltojson/#.UjbdxMbIahs>
- <http://www.freeformatter.com/json-to-xml-converter.html>
- <http://www.freeformatter.com/xml-to-json-converter.html>

JQuery

- ¿Qué es JQuery?
 - Es una librería JavaScript
 - Permite trabajar de forma más sencilla que con el propio JavaScript
 - Proporciona independencia del navegador para el manejo del código JavaScript

JQuery

■ Ejemplo:

■ Ocultar divs con JavaScript puro

```
divs = document.getElementsByTagName('div');  
  
for (i = 0; i < divs.length; i++) {  
    divs[i].style.display = 'none';  
}
```

■ Ocultar divs con JQuery

```
$("div").hide();
```

JQuery

■ ¿Cómo se usa

<script src="http://code.jquery.com/jquery-1.10.2.min.js"></script>

```
<!DOCTYPE html>
<html>
  <head>
    ...
    <script src="http://code.jquery.com/jquery-1.10.2.min.js"></script>
    ...
  </head>

  <body>
    ...
  </body>
</html>
```

JQuery

- Filosofía:

1. Encontrar algún objeto HTML
2. Hacer algo con el objeto encontrado

JQuery

- Encontrar elementos: **SELECTOR**

- Se representa por **\$()**

`$("#myId")` `$(".myClass")` `$("#table")`

- **`$("#content")`** → objeto con atributo id
- **`$("li:first")`** → primer item de una lista
- **`$("tr:odd")`** → filas impares de una tabla
- **`$("a[target=_blank]")`** → enlaces con valor de target a "_blank"
- **`$("form[id^=step]")`** → formularios cuyo id comienza con "step"

JQuery

- SELECTOR. Es posible realizar selecciones multiples:

`$("#myId, .myClass, table")`

- ACTUAR. Es tan simple como invocar al método correspondiente sobre el objeto seleccionado

`$("div").addClass("redbox");`

JQuery

- Más características:

- Encadenar métodos:

- `$("div").addClass("redbox").fadeOut();`**

- Un método puede tener muchos usos:

- `$("...").html();`**

- `$("...").html("<p>hello</p>");`**

- `$("...").html(function(i){
 return "<p>hello " + i + "</p>";
});`**

JQuery - Métodos

- Mover elementos
append(), appendTo(), before(), after(), ...
- Atributos
css(), attr(), html(), val(), addClass(), ...
- Eventos
bind(), trigger(), unbind(), click(), ...
- Efectos
show(), fadeOut(), toggle(), animate(), ...
- Traversing (recorrer)
find(), is(), prevAll(), next(), hasClass(), ...
- Ajax
get(), getJSON(), post(), ajax(), load(), ...

JQuery - Funciones

- Es posible pasar a JQuery una función para que se ejecute cuando se ha cargado la página:

`$(function(){`

**el código se ejecuta aquí cuando el DOM está
preparado**

`})`

- Es equivalente a:

`$(document).ready(function(){...})`

jQuery

■ Ejemplo de mover elementos

`$("#foo").append("<p>test</p>");`

```
<html>
<body>
<div>jQuery</div>
<div id="foo">example</div>
</body>
</html>
```

```
<html>
<body>
<div>jQuery</div>
<div id="foo">example<p>test</p></div>
</body>
</html>
```

jQuery

■ Ejemplo de mover elementos

`$("p").appendTo("#foo");`

```
<html>
<body>
<div>jQuery
  <p>moving</p>
  <p>paragraphs</p>
</div>
<div id="foo">example</div>
</body>
</html>
```

```
<html>
<body>
  <div>jQuery</div>
<div id="foo">example
  <p>moving</p>
  <p>paragraphs</p>
</div>
</body>
</html>
```

jQuery - Atributos

■ Get

.attr('id')



.html()



.val()



.css("top")



.width()



■ Set

.attr('id','foo')

.html("<p>hi</p>")

.val("new val")

.css("top","80px")

.width(60)

JQuery - Atributos

■ Ejemplos:

- Borde negro de 1px

```
$(...).css("border","1px solid black");
```

- Varias propiedades simultáneas

```
$(...).css({  
    "background":"yellow",  
    "height":"400px"  
});
```

- Configurar atributos href de todos los enlaces

```
$("a").attr("href","http://www.uv.es");
```

JQuery - Atributos

■ Ejemplos:

- Reemplazar código html

`$(...).html("<p>Good Bye</p>");`

```
<div>Hello</div>
```



```
<div><p>Good Bye</p></div>
```

- Poner un check a checked

`$(":checkbox").attr("checked","checked");`

- Configurar un valor a 5

`$(...).val("5");`

jQuery - Eventos

- Hacer algo al presionar un botón:

```
$("#button").click(function(){  
    algo();  
});
```

- Configurar un evento y lanzarlo:

```
$("#button").bind("expand", function(){  
    algo();  
});  
$("#button:first").trigger("expand");
```

- Desenlazar el evento creado:

```
$("#button").unbind("expand");
```

jQuery - Eventos

- Asignar eventos al documento (live deprecated):

```
$("#button").on('click',function(){  
    algo();  
});
```

- Asignar delegación de eventos a elementos (delegate deprecated):

```
$("#form").on("click", "button", function(){  
    algo();  
});
```

jQuery - Animaciones

- Mostrar y Ocultar con movimiento

```
$(...).click(function(){  
    $("div:first").slideToggle();  
});
```

- Animar elementos a un tamaño en un tiempo

```
$(...).animate({"width":"300px"},500);
```

- Eliminar el foco haciendo un "fading" al 30% de opacidad en 0.5s

```
$(...).fadeTo(500,0.3);
```


JQuery – Recorridos (traversing)

- Obtener las celdas anteriores a #myCell

`$("#myCell")`

```
<html>
<body>
  <table><tr>
    <td></td>
    <td></td>
    <td id="myCell"></td>
    <td></td>
  </tr></table>
</body>
</html>
```

jQuery – Recorridos (traversing)

- Obtener las celdas anteriores a #myCell

`$("#myCell").prevAll();`

```
<html>
<body>
  <table><tr>
    <td></td>
    <td></td>
    <td id="myCell"></td>
    <td></td>
  </tr></table>
</body>
</html>
```

jQuery – Recorridos (traversing)

- Obtener las celdas anteriores a #myCell

`$("#myCell").prevAll().andSelf();`

```
<html>
<body>
  <table><tr>
    <td></td>
    <td></td>
    <td id="myCell"></td>
    <td></td>
  </tr></table>
</body>
</html>
```


jQuery – AJAX

- Cargar datos del servidor usando un HTTP GET
jQuery.get()
- Cargar datos JSON del servidor usando un HTTP GET
jQuerygetJSON()
- Cargar datos del servidor usando un HTTP POST
jQuery.post()
- Cargar un fichero JavaScript del servidor usando un HTTP GET, y ejecutarlo posteriormente
jQuery.getScript()
- Cargar datos del servidor y situar el contenido HTML recibido en un determinado elemento
.load()

<http://api.jquery.com/category/ajax/>

JQuery – AJAX (ejemplos)

- Hacer una llamada AJAX sin procesar la respuesta
`$.get("pagina.jsp",{ "param": "valor" });`
- Hacer una llamada AJAX y procesar la respuesta
**`$.post("pagina.jsp",{ "param": "valor" },
function(respuesta){
 alert(respuesta);
});`**
- Cargar datos con load
**`$("#feeds").load("feeds.php", {limit: 25},
function(){
 alert("The last 25 entries have been loaded");
});`**

JQuery – Funciones

- Para declarar funciones JQuery se utiliza la palabra reservada "fn" de función:

```
$.fn.tufuncion = function (parametros) {  
    // el resto de instrucciones  
}
```

- La invocación a la función es como una llamada a cualquier otra función de JQuery:

```
$("#id").tufuncion("parametros si existen");
```

o bien con una llamada genérica:

```
$().tufuncion("parametros si existen");
```

JQuery – Funciones – Ejemplo

```
//Creamos una función JQuery para cambiar el background del elemento
$.fn.background = function(_background) {
    $(this).css({background: _background});
}

//Creamos una función para animar el elemento
// que aceptará un valor booleano true = si animar | false = no animar
$.fn.animar = function(_bool) {
    if (_bool == true) {
        $(this).css({position: "relative"});
        $(this).animate({left: "+=200", bottom: "+=100"}, 1000, function()
        {
            $(this).animate({left: "-=200", bottom: "-=100"}, 1000);
        });
    }
}

//Ahora llamamos a las nuevas funciones JQuery
$(function(){
    $("#demostracion").background("black");
    $("#crear_efecto").click(function(){
        $("#demostracion").animar(true);
    });
});

...

<input id="crear_efecto" type="button" value="Crear efecto">
<div id="demostracion" style="width: 250px; height: 250px;">5</div>
```

JQuery – Funciones

- Lo habitual es usar las funciones definidas en jQuery como métodos de un objeto, puesto que en realidad fn no es más que un alias de la propiedad "prototype".
- Se pueden encadenar llamadas si la función definida devuelve el objeto con el que está trabajando, por ejemplo:

```
$.fn.blueBorder = function(){  
    this.each(function(){  
        $(this).css("border","solid blue 2px");  
    });  
    return this;  
};  
$.fn.blueText = function(){  
    this.each(function(){  
        $(this).css("color","blue");  
    });  
    return this;  
};  
$("#b").click(function(){  
    $(' .blue').blueBorder().blueText();  
});
```

```
<input id="b" type="button"  
    value="Apply" /><br />  
<div class="blue">1</div>  
<div class="blue">2</div>  
<div class="blue">3</div>
```

JQuery – Caching + Chaining

- Cada vez que se realiza una acción, el selector interno de JQuery (Sizzle) realiza un recorrido completo del DOM hasta localizar los objetos
- Esto es ineficiente y se puede evitar cacheando los objetos a través de una asignación a variables JavaScript (que se denotan con un \$ delante para distinguirlas del resto)
- Adicionalmente, se pueden encadenar secuencias de acciones (lo que comúnmente se conoce como chaining), lo que permite realizar varias tareas de forma simultánea en la misma instrucción.

JQuery – Caching + Chaining

```
$('#myObjId').click(function() {  
    if ( $('#myObjId').hasClass('clicked') ) {  
        $('#myObjId').removeClass('clicked');  
        $('#myObjId').css('background-color', 'red');  
    } else {  
        $('#myObjId').addClass('clicked');  
        $('#myObjId').css('background-color', 'blue');  
    }  
});  
$('#myObjId').val('Click Me!');
```



```
var $myObj = $('#myObjId');  
$myObj.click(function(){  
    if ( $myObj.hasClass('clicked') ){  
        $myObj.removeClass('clicked').css('background-color', 'red');  
    } else {  
        $myObj.addClass('clicked').css('background-color', 'blue');  
    }  
});  
$myObjId.val('Click Me!');
```

JQuery

■ Referencias

■ Introducción a JQuery

- <http://jquery.com/>

■ API

- <http://api.jquery.com>

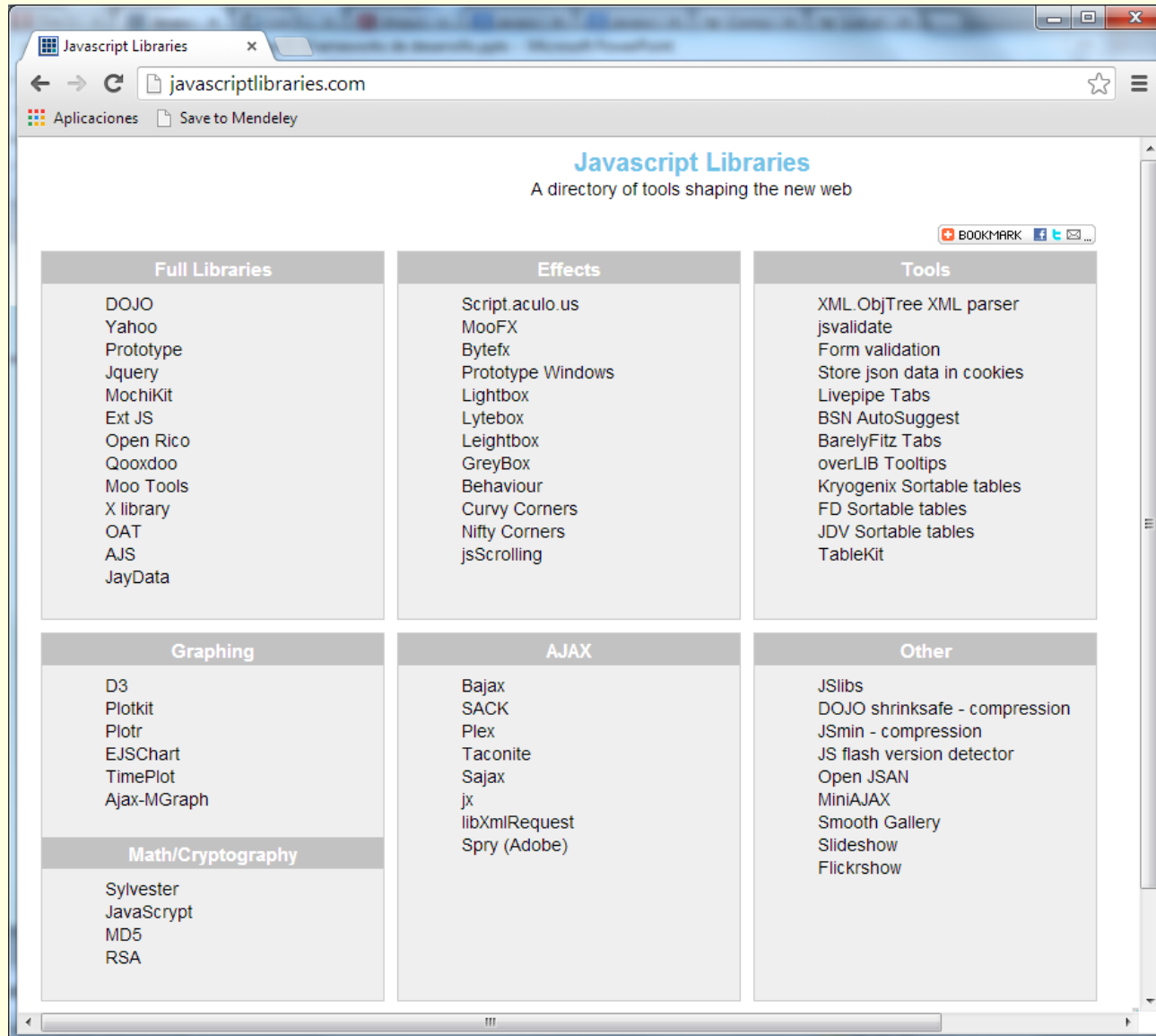
■ Libros gratuitos:

- JQuery CookBook by the community of experts
- JQuery Succintly by Cody Lindley

■ Ejemplos

- <http://www.jqueryrain.com/>

Otros Frameworks de JavaScript



Otros Frameworks de JavaScript

■ Ext JS

<http://www.sencha.com/products/extjs/>

The screenshot displays the Ext JS Kitchen Sink application in a web browser. The application has a green header bar with the title "Ext JS Kitchen Sink" and a "Theme: Neptune" dropdown. The main content area is divided into three panels: "Examples", "Progress Bar Pager", and "Example Info".

The "Examples" panel on the left shows a tree view of various components. The "Progress Bar Pager" example is selected, and its content is displayed in the central panel. It features a table with the following data:

Company	Price	Change	% Change	Last Updated
3m Co	\$71.72	0.02	0.03%	09/01/2013
Alcoa Inc	\$29.01	0.42	1.47%	09/01/2013
Altria Group Inc	\$83.81	0.28	0.34%	09/01/2013
American Express Company	\$52.55	0.01	0.02%	09/01/2013
American International Group, Inc.	\$64.13	0.31	0.49%	09/01/2013
AT&T Inc.	\$31.61	-0.48	-1.54%	09/01/2013
Boeing Co.	\$75.43	0.53	0.71%	09/01/2013
Caterpillar Inc.	\$67.27	0.92	1.39%	09/01/2013

Below the table, there is a pagination control showing "Page 1 of 3" and "Displaying 1 - 10 of 29".

The "Example Info" panel on the right contains a "Description" section stating: "This example demonstrates using a custom paging display." Below this is a "Code Preview" section showing the Ext JS code used for the example:

```
Ext.define('KitchenSink.view.  
    extend: 'Ext.grid.Panel',  
  
    requires: [  
        'Ext.data.*',  
        'Ext.grid.*',  
        'Ext.util.*',  
        'Ext.toolbar.Paging',  
        'Ext.ux.ProgressBarPa  
        'KitchenSink.model.Co  
    ],  
    xtype: 'progress-bar-page  
  
    stripeRows: true,  
    height: 320,  
    frame: true,  
    title: 'Progress Bar Page  
  
    initComponents: function()  
        this.width = 650;
```


Otros Frameworks de JavaScript

■ DOJO

<http://dojotoolkit.org/>

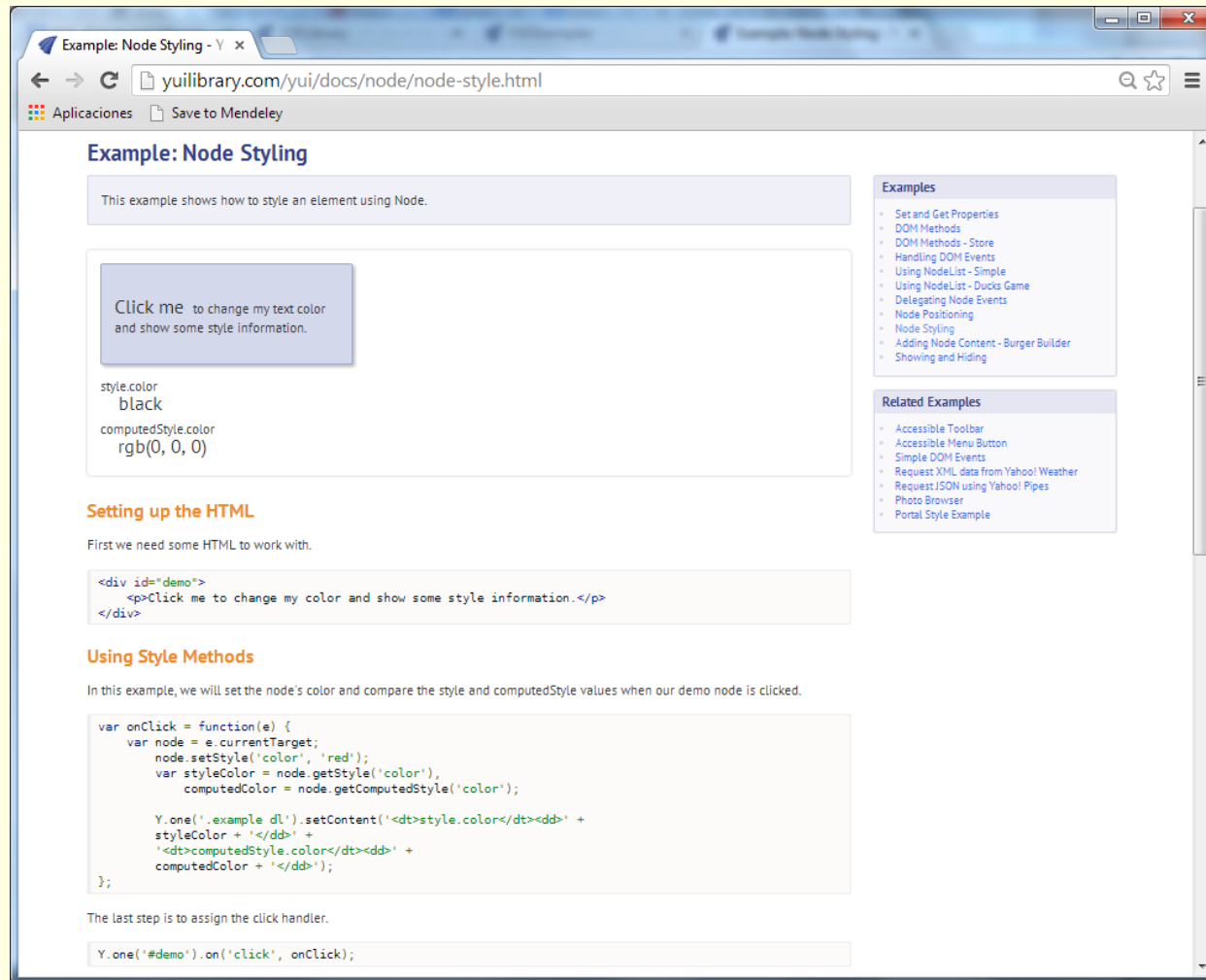
The screenshot shows a web browser window titled "The Dojo Toolkit - Demos" with the URL "demos.dojotoolkit.org/demos/". The page is titled "Graphics & Charting" and displays a grid of 12 interactive demos, each with a thumbnail image, a title, and a brief description.

Demo Title	Description
Geo Charting	A sample application showing demographic data bound to Geo Chart widgets. Illustrates mobile spin wheel and touch/mouse event handling as well as orientation changes. Works desktop and mobile.
Mobile Stock Charting	This application displays three stock charts. It works well for both desktop and mobile devices. On mobile devices, try two-finger touching to display the difference between quote values and touch-scroll across dates.
Shipping Routes Map	Map nodes and links for shipping flights between cities geographically. This demo makes use of Open Layers and GFX shape integration.
Day/Night Map	Show day/night regions with data plots for various worldwide cities. This demo makes use of Open Layers and GFX shape integration.
Map Tile Providers	This demo shows allows you to swap out different map tile providers using Dojo and OpenLayers integration.
Mobile Gauges	Play around with features of gauges using this gauge feature explorer. Desktop and Mobile (touch) enabled.
Mobile Maps	Simple Mobile mapping demo, making use of OpenLayers and Dojo Mobile for navigation.
Chart Plot Types	View each of Dojo charting's main plot types in different out-of-box themes.
Spider Chart	This demo lets you try various features of the Dojo spider chart plot type.
Chart Dynamics	This demo lets you try various dynamically changeable features of charts.
Pie Chart Smart Labels	This demo shows smart labelling features for pie chart labels.
Clock	A simple interactive clock. Try grabbing the hands to change the time. Built with Dojo's graphics library.

Otros Frameworks de JavaScript

■ YUI

<http://yuilibrary.com/>



The screenshot shows a web browser window displaying the 'Example: Node Styling' page from the YUI library documentation. The browser's address bar shows the URL 'yuilibrary.com/yui/docs/node/node-style.html'. The page content includes a title 'Example: Node Styling', a description 'This example shows how to style an element using Node.', and a button labeled 'Click me' with the text 'to change my text color and show some style information.' Below the button, the current style and computed style are shown: 'style.color' is 'black' and 'computedStyle.color' is 'rgb(0, 0, 0)'. The page also features sections for 'Setting up the HTML' with the required HTML code, 'Using Style Methods' with the JavaScript code for the click handler, and a list of 'Examples' and 'Related Examples' on the right side.

Example: Node Styling

This example shows how to style an element using Node.

Click me to change my text color and show some style information.

style.color
black
computedStyle.color
rgb(0, 0, 0)

Setting up the HTML

First we need some HTML to work with.

```
<div id="demo">
  <p>Click me to change my color and show some style information.</p>
</div>
```

Using Style Methods

In this example, we will set the node's color and compare the style and computedStyle values when our demo node is clicked.

```
var onClick = function(e) {
  var node = e.currentTarget;
  node.setStyle('color', 'red');
  var styleColor = node.getStyle('color'),
      computedColor = node.getComputedStyle('color');

  Y.one('#example dl').setContent('<dt>style.color</dt><dd>' +
    styleColor + '</dd>' +
    '<dt>computedStyle.color</dt><dd>' +
    computedColor + '</dd>');
};
```

The last step is to assign the click handler.

```
Y.one('#demo').on('click', onClick);
```

Examples

- Set and Get Properties
- DOM Methods
- DOM Methods - Store
- Handling DOM Events
- Using NodeList - Simple
- Using NodeList - Ducks Game
- Delegating Node Events
- Node Positioning
- Node Styling
- Adding Node Content - Burger Builder
- Showing and Hiding

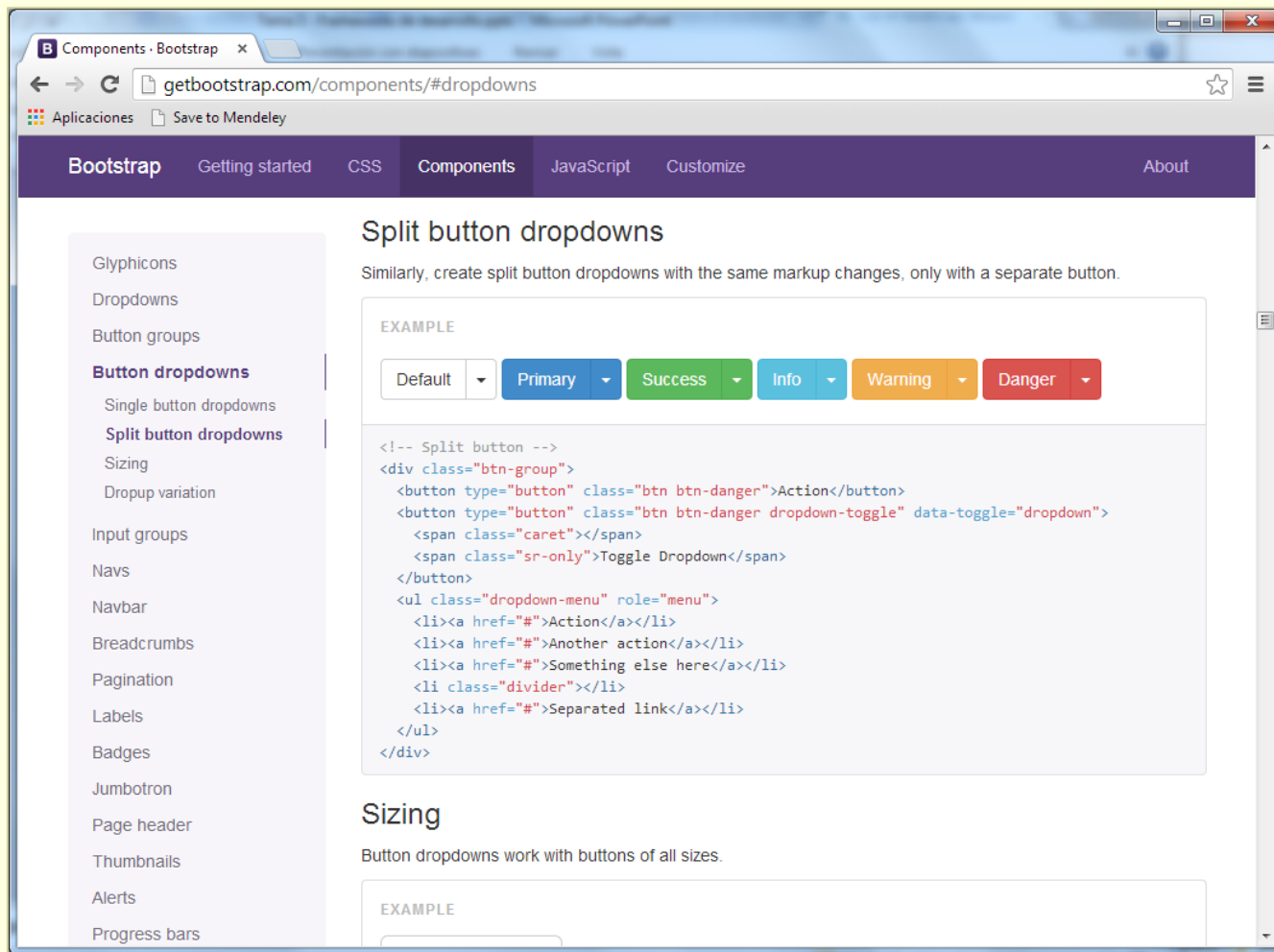
Related Examples

- Accessible Toolbar
- Accessible Menu Button
- Simple DOM Events
- Request XML data from Yahoo! Weather
- Request JSON using Yahoo! Pipes
- Photo Browser
- Portal Style Example

Otros Frameworks de JavaScript

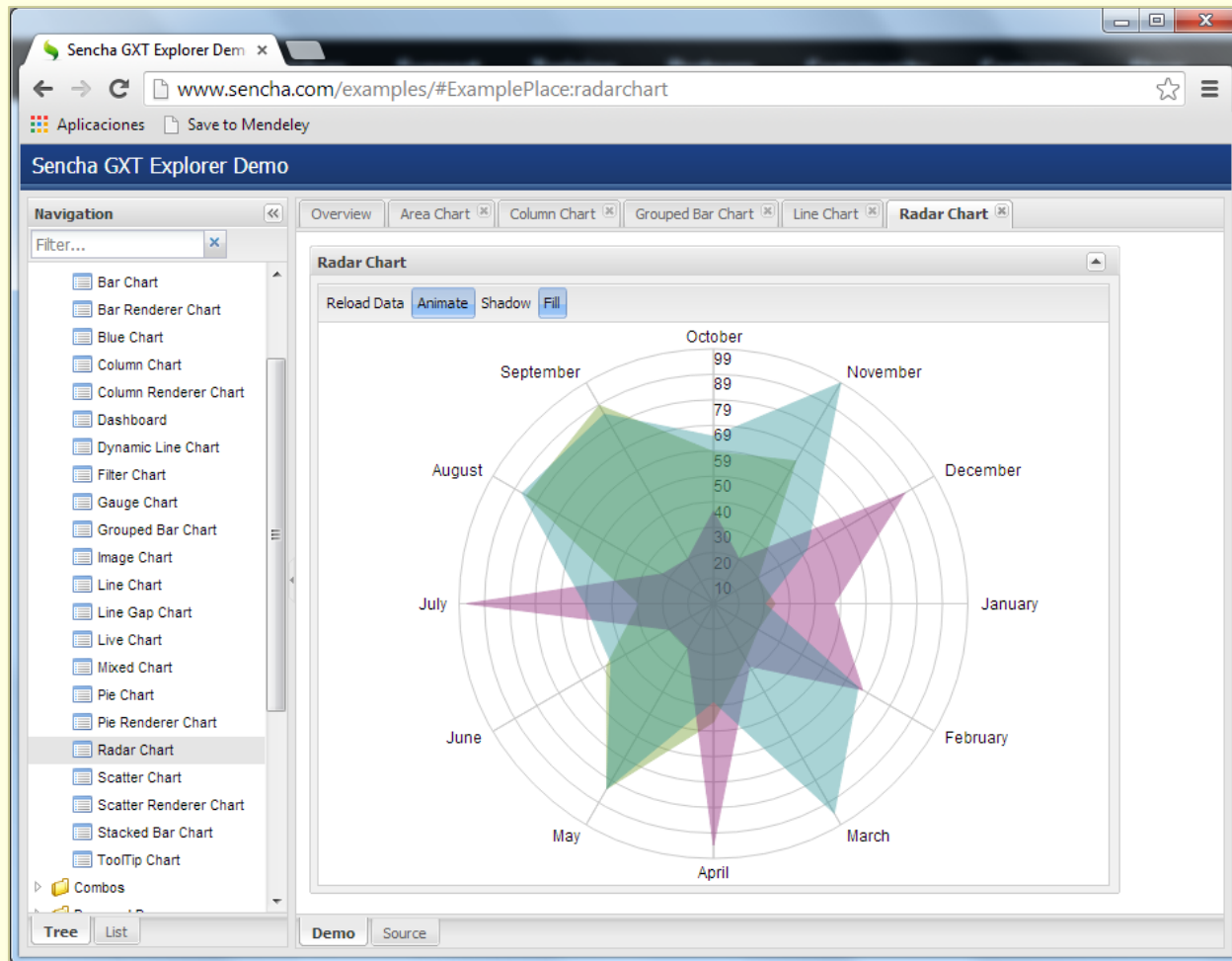
■ BootStrap

<http://getbootstrap.com/>



Otros Frameworks de JavaScript

- Sencha GXT <http://www.sencha.com/products/gxt/>



Otros Frameworks de JavaScript

■ MVC frameworks

■ Backbone.js



<http://documentcloud.github.io/backbone/>

■ Ember.js



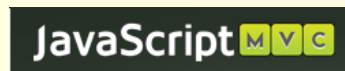
<http://emberjs.com/>

■ Knockout.js



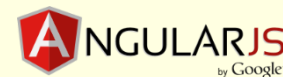
<http://knockoutjs.com/>

■ Javascript MVC



<http://javascriptmvc.com/>

■ Angular.js



<http://angularjs.org/>

Otros Frameworks de JavaScript

■ Un mundo interminable...

<http://microjs.com/#>

