

# Tema 2: Tecnologías para el desarrollo en el cliente

HTML y XHTML

CSS

JavaScript

Sistemas de Información para Internet  
3º del Grado de Ingeniería Informática (tres menciones)

J. Francisco Chicano

Departamento de Lenguajes y Ciencias de la Computación  
Escuela Técnica Superior de Ingeniería Informática  
Universidad de Málaga

# JavaScript



# JavaScript

## Introducción

- JavaScript es una implementación de la especificación de lenguaje **ECMAScript**
- JavaScript fue desarrollado por Brendan Eich (Netscape) con la idea de potenciar la creación de páginas Web dinámicas para **Netscape Navigator**
- A pesar del nombre, **JavaScript** se parece al lenguaje Java sólo en su sintaxis

*"It was all within six months from May till December (1995) that it was Mocha and then LiveScript. And then in early December, Netscape and Sun did a license agreement and it became JavaScript. And the idea was to make it a complementary scripting language to go with Java, with the compiled language" (B. Eich)*

- Permite insertar comportamiento en las páginas Web, ampliando las posibilidades de **HTML** y **CSS**
- JavaScript es interpretado por el navegador (agente de usuario)

# JavaScript

## Inclusión en documento (X)HTML

- Para hacer que un documento HTML incluya código JavaScript se debe hacer uso del elemento SCRIPT de esta forma:

```
<script type="text/javascript">  
  código JavaScript  
</script>
```

- En XHTML el código JavaScript se escribe:

```
<script type="text/javascript">  
<![CDATA[  
  código JavaScript  
]]>  
</script>
```

# JavaScript

## Inclusión en documento (X)HTML

- También se puede utilizar el código JavaScript escrito en un archivo separado (recomendado)
- Este archivo suele tener la extensión `js`
- Después ese código se puede enlazar desde la página web así:

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

# JavaScript

## Sintaxis

- La sintaxis se hereda de los lenguajes “C-like”
- Los operadores y estructuras de control son (casi) los mismos que en Java, C o C++
- Los comentarios también son iguales que en estos lenguajes
- Características:
  - Comentarios de línea: //
  - Bloque de comentarios: /\* \*/
  - Sensible a mayúsculas
  - Punto y coma tras cada sentencia
  - Estructuras de control: if, for, while, do while, switch
  - Bloques de código: { }
  - Llamada a funciones: fn (3,4)

# JavaScript

## Variables

- JavaScript no es un lenguaje fuertemente tipado: cualquier variable puede almacenar cualquier tipo de datos y cambiar su tipo durante la ejecución
- No es necesario declarar variables (si no se hace se asume que la variable es global)
- Para declarar una variable escribimos:


```
var variable = valor;
```

- Ejemplos:

```
var testar = 0;  
testaTexto = "Mi casa";  
SeleccionarColor = true;
```

# JavaScript

## Tipos

- Los datos almacenados en una variable pueden ser:
  - Número
  - Cadena
  - Booleano
  - Función  Esto supone una diferencia con respecto a C++ y Java
  - Objeto
- La representación de los números, Booleanos y cadenas es la habitual con la salvedad de que las cadenas pueden ir delimitadas entre comillas simples
- Una función se define de la siguiente forma:

```
function ctr()  
{  
    this.ctr="a";  
    if (this == window)  
    {  
        alert("si");  
    }  
}
```



# JavaScript

## Tipos

- Los objetos se pueden crear de tres formas: literal, operador **new** y método **create**

```
var an = new Animal ();  
var obj = Object.create();  
var json = { nombre: "Antonio", apellido: "García" };
```

- En JavaScript no existe el concepto de clase tal cual lo conocemos
- Los “constructores” y los métodos no son más que funciones

```
function Animal ()  
{  
    this.nombre = "tigre";  
    this.peso = 100;  
    this.getPeso = function () {return this.peso; };  
}
```

- Por este motivo algunos programadores describen a los objetos en JavaScript como “property bags”

# JavaScript

## Diálogos

- Usando JavaScript es posible mostrar al usuario diálogos con mensajes, petición de datos o solicitud de confirmaciones
- **alert** es una función que nos permite mostrar un mensaje al usuario
- Su sintaxis es:

```
alert(texto_del_mensaje);
```

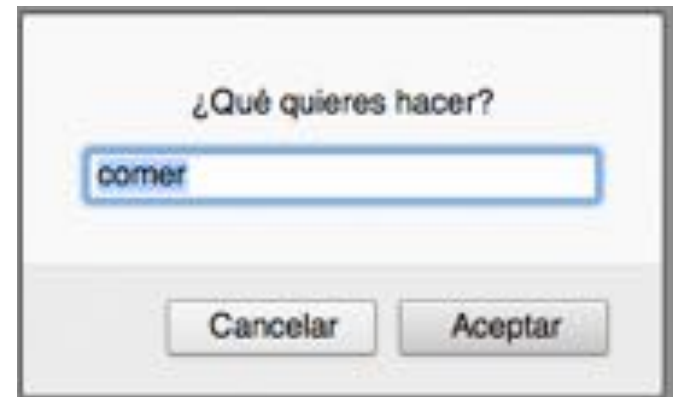


# JavaScript

## Diálogos

- Para pedir un dato concreto al usuario podemos usar:  
`prompt(texto_del_mensaje,valor_por_defecto);`
- El segundo parámetro (valor por defecto) no es obligatorio y permite asignar un valor al cuadro de texto en el que el usuario tendrá que introducir información.
- El cuadro de diálogo que saca `prompt` posee dos botones, uno es el de Aceptar y el otro es el de Cancelar. Si el usuario pulsa Cancelar, la función `prompt` devuelve el valor nulo (null).

```
var a = prompt("¿Qué quieres hacer?", "comer");
```



# JavaScript

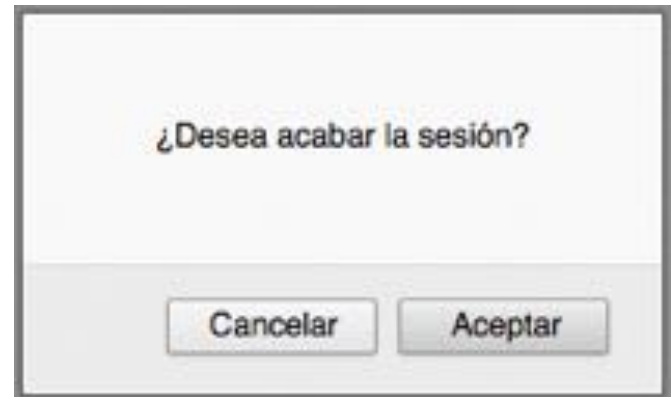
## Diálogos

- Si lo único que hace falta del usuario es una respuesta binaria es posible usar `confirm`:

```
confirm(texto_del_mensaje);
```

- La ventana mostrará el texto elegido (normalmente es una pregunta) y el usuario elegirá si desea aceptar o no el contenido.
- `confirm` devuelve un valor `true` en el caso de que el usuario acepte el mensaje, y `false` si no lo hace

```
var a = confirm ("¿Desea acabar la sesión?");
```



# JavaScript

## Objeto window

- Este objeto representa a la ventana en la cual se ve la página web.
- En el caso de que la página tenga marcos, se generan tantos objetos `window` como marcos haya
- Es el objeto del que debemos partir para manipular la página Web

### Window Object Properties

Property	Description
<code>closed</code>	Returns a Boolean value indicating whether a window has been closed or not
<code>defaultStatus</code>	Sets or returns the default text in the statusbar of a window
<code>document</code>	Returns the Document object for the window ( <a href="#">See Document object</a> )
<code>frames</code>	Returns an array of all the frames (including iframes) in the current window
<code>history</code>	Returns the History object for the window ( <a href="#">See History object</a> )
<code>innerHeight</code>	Returns the inner height of a window's content area
<code>innerWidth</code>	Returns the inner width of a window's content area
<code>length</code>	Returns the number of frames (including iframes) in a window
<code>location</code>	Returns the Location object for the window ( <a href="#">See Location object</a> )
<code>name</code>	Sets or returns the name of a window
<code>navigator</code>	Returns the Navigator object for the window ( <a href="#">See Navigator object</a> )

# JavaScript

## Objeto window

<code>opener</code>	Returns a reference to the window that created the window
<code>outerHeight</code>	Returns the outer height of a window, including toolbars/scrollbars
<code>outerWidth</code>	Returns the outer width of a window, including toolbars/scrollbars
<code>pageXOffset</code>	Returns the pixels the current document has been scrolled (horizontally) from the upper left corner of the window
<code>pageYOffset</code>	Returns the pixels the current document has been scrolled (vertically) from the upper left corner of the window
<code>parent</code>	Returns the parent window of the current window
<code>screen</code>	Returns the Screen object for the window ( <a href="#">See Screen object</a> )
<code>screenLeft</code>	Returns the x coordinate of the window relative to the screen
<code>screenTop</code>	Returns the y coordinate of the window relative to the screen
<code>screenX</code>	Returns the x coordinate of the window relative to the screen
<code>screenY</code>	Returns the y coordinate of the window relative to the screen
<code>self</code>	Returns the current window
<code>status</code>	Sets or returns the text in the statusbar of a window
<code>top</code>	Returns the topmost browser window

# JavaScript

## Objeto window

### Window Object Methods

Method	Description
<code>alert()</code>	Displays an alert box with a message and an OK button
<code>atob()</code>	Decodes a base-64 encoded string
<code>blur()</code>	Removes focus from the current window
<code>btoa()</code>	Encodes a string in base-64
<code>clearInterval()</code>	Clears a timer set with <code>setInterval()</code>
<code>clearTimeout()</code>	Clears a timer set with <code>setTimeout()</code>
<code>close()</code>	Closes the current window
<code>confirm()</code>	Displays a dialog box with a message and an OK and a Cancel button
<code>createPopup()</code>	Creates a pop-up window
<code>focus()</code>	Sets focus to the current window
<code>moveBy()</code>	Moves a window relative to its current position
<code>moveTo()</code>	Moves a window to the specified position
<code>open()</code>	Opens a new browser window

# JavaScript

## Objeto window

<code>print()</code>	Prints the content of the current window
<code>prompt()</code>	Displays a dialog box that prompts the visitor for input
<code>resizeBy()</code>	Resizes the window by the specified pixels
<code>resizeTo()</code>	Resizes the window to the specified width and height
<code>scroll()</code>	This method has been replaced by the <code>scrollTo()</code> method.
<code>scrollBy()</code>	Scrolls the content by the specified number of pixels
<code>scrollTo()</code>	Scrolls the content to the specified coordinates
<code>setInterval()</code>	Calls a function or evaluates an expression at specified intervals (in milliseconds)
<code>setTimeout()</code>	Calls a function or evaluates an expression after a specified number of milliseconds
<code>stop()</code>	Stops the window from loading



# JavaScript

## Objeto document

- Representa al documento HTML
- Debe implementar la interfaz especificada por el Document Object Model (DOM)
  - <http://www.w3.org/DOM/>
- La versión activa es la DOM3 (desde 2004) pero se está trabajando en la DOM4
- Este objeto permite navegar y alterar cualquier elemento del documento (X)HTML
- Por ejemplo:

```
document.forms[0]           // primer formulario de la página
document.images[3]          // cuarta imagen de la página
document.images.logo         // imagen con ID="logo"
document.getElementById("pie") // elemento con ID="pie"
```

# JavaScript

## Objeto document

- Un caso habitual es usar JavaScript para validar los campos de un formulario
- Para acceder a un formulario podemos usar el índice del formulario o su nombre

```
<form name="formulario">  
  <input type="text" name="entrada" size="20"/>  
  <button onclick="valida()">Pulsame</button>  
</form>
```

- Si queremos acceder al contenido del campo de texto:

```
var a = document.formulario.entrada.value;
```

# JavaScript

## Buscando en el DOM

- Buscando por ID
  - `document.getElementById("campo");`
- Buscando por nombre
  - `document.getElementsByName("lang");`
- Buscando por clase
  - `document.getElementsByClassName("miClase");`
- Buscando por etiqueta
  - `document.getElementsByTagName("li");`

# JavaScript

## Modificando el DOM

- Podemos modificar el contenido de un elemento
  - `document.getElementById("contenido").innerHTML = "Esto es un contenido de JavaScript";`
- El archivo de una imagen
  - `document.getElementById("miImagen").src = "nueva_imagen.jpg";`
- Los estilos de un elemento
  - `document.getElementById("element").style.display = "none";`
- Añadir una clase
  - `document.getElementById("element").classList.add("miClase");`
- Eliminar un elemento (y todo su contenido)
  - `document.getElementById("element").innerHTML = "";`

# JavaScript

## Eventos

- Un evento es un suceso que ocurre cuando el usuario realiza alguna acción (pasa el ratón por encima de un objeto de la página, pulsa una tecla, pulsa un botón, envía un formulario, etc.)
- Algunos eventos los produce el navegador (la carga de la página)
- Los eventos se colocan en etiquetas de (X)HTML, de esta forma:

`<etiquetaHTML atributos... nombreDeEvento="expresiónJavaScript">`

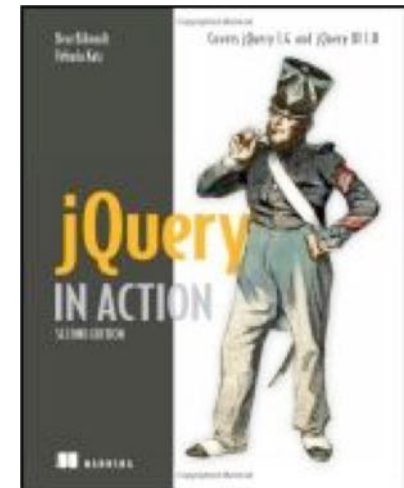
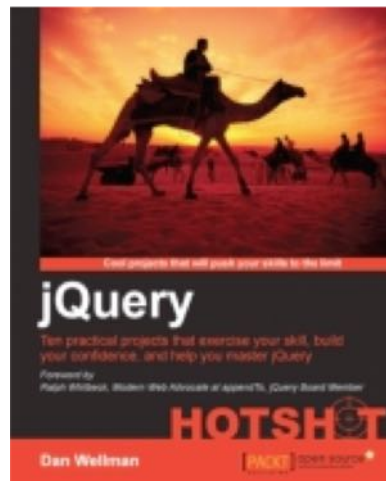
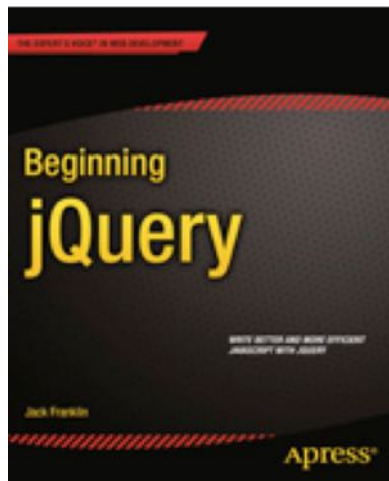
- De tal forma que cuando se produce el evento en cuestión, se ejecuta el código en JavaScript que está entre comillas.

```
<a href="cities.html" onclick="alert('Pulsaste!');">Enlace</a>
```

- Lista de eventos:
  - [http://www.w3schools.com/tags/ref\\_eventattributes.asp](http://www.w3schools.com/tags/ref_eventattributes.asp)

# JavaScript jQuery

- **jQuery** es una biblioteca que facilita la programación en JavaScript
- Puede encontrarse en <http://jquery.com>
- La versión actual es la 3.3.1
- Es posible encontrar libros completos sobre esta biblioteca



# jQuery

## Selectores

- Buscando por ID
  - `$("#id");`
- Buscando por clase
  - `$(".miClase");`
- Buscando por etiqueta
  - `$("p");`
- Búsqueda avanzada
  - `$("p.miClase span.rojo");`

# jQuery

## Eventos

- click
  - `$(".clickable").click(function(){  
    alert("Click!");  
});`
- change
  - `$(".select").change(function(){  
    var str = "";  
    $(".select option:selected").each(  
        function() { str += $( this  
    ).text() + " "; }); });`
- Más eventos en:  
<https://api.jquery.com/category/events/>



# JavaScript

## Angular

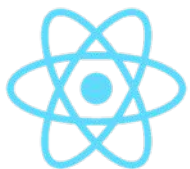
- **AngularJS** es un *framework* desarrollado por Google basado en el patrón Modelo – Vista – Controlador
- A partir de la versión 2.0 cambió a Angular (en vez de AngularJS)
- Angular 2 utiliza TypeScript
  - TypeScript es un lenguaje de programación libre y de código abierto desarrollado y mantenido por Microsoft. Es un superconjunto de JavaScript, que añade tipado estático y objetos basados en clases. Compila en JavaScript original.
- Se utiliza para crear *front-ends* de aplicaciones Web (con un back-end formado por un servicio Web REST)
- También se utiliza para programa *Single Page Applications* (SPA)
- Referencias:
  - [Angularjs.org](https://angularjs.org)
  - [Angular.io](https://angular.io)
  - [typescriptlang.org](https://typescriptlang.org)



# JavaScript

## ReactJS

- ReactJS es un *framework* desarrollado por Facebook para simplificar la creación de interfaces de usuario
- <https://reactjs.org/>
- ```
function formatName(user) {  
    return user.firstName + ' ' + user.lastName;  
}
```
- ```
const user = {  
    firstName: 'Harper',  
    lastName: 'Perez'  
};
```
- ```
const element = ( <h1>    Hello, {formatName(user)}!  </h1> );
```
- ```
ReactDOM.render( element, document.getElementById('root'));
```



React

# JavaScript

## ReactJS

- JSX es una extensión de ECMAScript (aka: JavaScript o JS) sin definición semántica y que tiene una sintaxis muy parecida al XML. **No existe intención alguna de implementarlo en motores o navegadores y menos aún tomarlo como una propuesta para ser incorporada en la especificación de ECMAScript.** Fue creado con la intención de que los preprocesadores (ej. [Babel.js](#)) lo transformen en ECMAScript estándar.
- `// JSX`  
`<div prop="miProp"> <h1>Soy un Header!</h1> </div>`
- `// JS`  
`React.createElement( "div", { prop: "miProp" },`  
`React.createElement( "h1", null, "Soy un Header!" ) );`

# JavaScript

## NodeJS

- JavaScript del Lado del Servidor
- Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome
- <https://nodejs.org/es/>

```
■ var http = require("http");  
■ var url = require("url");  
■ function onRequest(request, response) {  
■     var pathname = url.parse(request.url).pathname;  
■     console.log("Petición para " + pathname + " recibida.");  
■     response.writeHead(200, {"Content-Type": "text/html"});  
■     response.write("Hola Mundo");  
■     response.end();  
■ }  
  
■ http.createServer(onRequest).listen(8888);  
■ console.log("Servidor Iniciado.");
```



# JavaScript

## NodeJS

- Express: Framework para Node.js para aplicaciones web, enfocado al middleware
- ```
const express = require('express')  
const app = express()  
const port = 3000  
app.get('/', function (req, res) {  
  res.send('Hello World!')  
})
```
- ```
app.post('/', function (req, res) {  
  res.send('Got a POST request')  
})
```
- ```
app.put('/user', function (req, res) {  
  res.send('Got a PUT request at /user')  
})
```
- ```
app.delete('/user', function (req, res) {  
  res.send('Got a DELETE request at /user')  
})
```
- ```
app.listen(port, () => console.log('Server listening on port ${port}!')
```

# Para ampliar conocimientos

- Tutorial de JavaScript:  
<http://www.w3schools.com/js/DEFAULT.asp>
- <https://developer.mozilla.org/es/docs/Web/JavaScript>
- Estándares de W3C sobre JavaScript  
[http://www.w3.org/standards/techs/js#w3c\\_all](http://www.w3.org/standards/techs/js#w3c_all)
- Estándares DOM  
<http://www.w3.org/DOM/DOMTR>
- Mark E. Daggett, "Expert JavaScript", Apress, 2013
- Adam Freeman, "Pro AngularJS", Apress, 2014
- Jack Franklin, "Beginning jQuery", Apress, 2013
- Antonio Cássio de Sousa, "Pro React", Apress, 2015

