Desarrollo Web En Entorno Cliente

TEMA 1

Arquitectura de desarrollo en el entorno cliente

Daniel López Lozano

Escuela Arte Granada

Índice

- Modelo de programación de aplicaciones Cliente-Servidor.
- Lenguajes asociados al entorno cliente.
- Introducción a JavaScript.

Modelo de programación de aplicaciones (Cliente – Servidor)

Daniel López Lozano

Escuela Arte Granada

- La **configuración arquitectónica** más habitual es la denominada **Cliente ⇔Servidor.**
- El cliente es un componente **consumidor de servicios**.
- El servidor es un proceso **proveedor de servicios**.
- Componente software que se utiliza en el cliente es el **navegador web**.
- **Permite acceder al contenido** ofrecido por los servidores de Internet **sin la necesidad** de que el usuario instale **un nuevo programa**.

- El **navegador web** permite a un usuario **acceder y visualizar a un recurso** publicado por un servidor Web a través de Internet y descrito mediante una dirección URL **(Universal Resource Locator).**
- **Mosaic.**
- **Netscape Navigator** (después Communicator).
- **Internet Explorer.**
- **Mozilla Firefox.**
- **Google Chrome.**
- **Safari.**
- **Opera.**

- El modelo de programación **Cliente⇔Servidor** se basa en dos elementos comunes:
- El lado del servidor(server-side): incluye el **hardware y software** del servidor Web así como diferentes elementos de **programación** y tecnologías incrustadas.
- Lenguajes como **PERL, Python, PHP, Java**, etc en la parte de la aplicación.
- Incluyendo también **tecnologías de servidor de bases de datos** que soporten sitios web.

- El lado del cliente(client-side): este elemento hace referencia a **los navegadores web** y está soportado por tecnologías como **HTML, CSS** y lenguajes como **JavaScript, Java Applets**, controles ActiveX y plugins.
- Se utilizan para **crear la presentación** de la página ó proporcionar **características interactivas**.
- Es justamente aquí dónde nos vamos a centrar a lo largo de todo el módulo.
- También es importante **el elemento de red** que proporciona protocolos de conectividad sobe cada lado.

Lenguajes asociados al entorno cliente

Daniel López Lozano

Escuela Arte Granada

- Dentro de la programación web existen dos grupos básicos de lenguajes de programación: **client-side y server-side**.
- En general las tecnologías client-side y server-side poseen características que las hacen complementarias más que adversarias.
- Por ejemplo, para recoger información de un **formulario y grabarla** en una base de datos, tendrá más sentido **chequear los datos** en el lado del cliente **antes de enviarlos** al servidor.

- La **programación en el lado del cliente** consigue que la validación del formulario sea mucho más eficiente y que el usuario se sienta menos frustrado al rellenar los datos en el formulario.
- Por otro lado **el almacenar los datos** en el servidor estaría mucho **mejor gestionado por una tecnología del lado del servidor** (server-side), dando por supuesto que la base de datos estará en el lado del servidor.

- Durante el curso vamos a usar **JavaScript porque es el lenguaje de script más utilizado** en el lado del cliente, y está soportado mayoritariamente por todas las plataformas.
- Por lo tanto a partir de ahora todas las referencias que hagamos al entorno cliente serán hacia JavaScript.

Tabla de las 4 capas del desarrollo web en el lado del cliente.	
Comportamiento (JavaScript)	
Presentación (CSS)	
Estructura (DOM / estructura HTML)	Contenido Estructurado (documento HTML)
Contenido (texto, imágenes, vídeos, etc.)	

- Como lenguaje del entorno cliente **JavaScript nos permite**:
- Conseguir que nuestra página web responda o reaccione directamente a la interacción del usuario con elementos de **formulario**, **enlaces de hipertexto o cualquier otro**.
- Controlar **múltiples ventanas, navegación y efectos basados** en las elecciones que ha hecho el usuario en el documento HTML.
- **Pre-procesar datos** en el cliente antes de enviarlos al servidor.

- **Modificar estilos y contenido** en los navegadores de forma dinámica e instantáneamente, en respuesta a interacciones del usuario.
- **Solicitar ficheros del servidor**, y enviar solicitudes de lectura y escritura a los lenguajes de servidor.
- **Versatilidad, velocidad de ejecución y flexibilidad** que hacen que no solo se use en aplicaciones web.
- Se encuentra integrado en aplicaciones de uso cotidiano como **OpenOffice, Adobe Acrobat, videojuegos**, etc

- Existen otras características, **relacionadas con la seguridad**, necesarias en el cliente:
- No se puede **modificar ó acceder a las preferencias** del navegador del cliente.
- No se puede lanzar la ejecución de una aplicación en el ordenador del cliente.
- Tampoco **leer/escribir ficheros/directorios** en el cliente.
- Las páginas web almacenadas **en diferentes dominios no pueden ser accesibles** por JavaScript.

Introducción a JavaScript

Daniel López Lozano

Escuela Arte Granada

- **Brendan Eich** que trabajaba en el navegador **Netscape** desarrollo un lenguaje de programación llamado **Mocha**, que posteriormente se denomino **LiveScript**.
- Coincidiendo con la popularidad del lenguaje Java acabo denominándose **Javascript**.
- ¿Entonces que es **ECMAScript**?
- Debido a que **Microsoft y otros sacaron sus propias versiones (JScript)**, los autores originales desarrollaron un estándar para la **ECMA** que es el adoptado en la actualidad por todos los navegadores.

- El objetivo de JavaScript **es manipular el código HTML**, mostrando, modificando y añadiendo elementos y su presentación.
- Todo ello para dotar de **interactividad al HTML**.
- **Existen 3 formas de incluir JavaScript** en nuestro código HTML:
- **La etiqueta <script>**
- **Ficheros externos js**
- **Incrustado en etiquetas HTML**

La etiqueta <script>

```
<head>
    <script type="text/javascript">
        alert("Hola Mundo");
    </script>
</head>
```

- Se pueden incluir todas las etiquetas script necesarias.
- Se recomienda hacerlo en la cabecera del documento.
- El **atributo type** es necesario.
- Se usa para incluir **pequeños trozos de código**.

Ficheros externos js

Fichero hmtl:

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

Fichero prueba.js:

```
alert("Hola mundo");
```

- Se simplifica **el código HTML de la página** y permite **reutilizar el mismo código** JavaScript en todas las páginas del sitio web.
- Es la forma **que utilizaremos nosotros** en clase.

Incrustado en etiquetas HTML

```
<body><p onclick="alert('Hola Mundo')">Texto de prueba.</p></body>
```

- Mezcla el **HTML con el JavaScript y complica el mantenimiento** del código de la web.
- Sólo se utiliza para definir **algunos eventos y en algunos otros** casos especiales.
- Esta forma no la vamos a utilizar en este modulo, **aunque es posible encontrarla en alguna aplicación**.

Etiqeta noscript

```
<noscript>
    <p>La página que estás viendo requiere para su funcionamiento el
        uso de JavaScript, por favor vuelva activarlo
    </p>
</noscript>
```

- Existe todavía la fama de que **JavaScript es inseguro** y bloquea el ordenador.
- Todavía se puede **desactivar de los navegadores**, por lo que es recomendable mostrar un aviso cuando ocurra.
- La etiqueta noscript debe incluirse en el body del HTML.

- La sintaxis de JavaScript en **la parte básica es similar a la de C, C++, Java** y otros lenguajes relacionados.
- No se tiene en cuenta **espacios en blanco** y saltos de línea.
- **Distingue** las mayúsculas y minúsculas.
- Se pueden incluir comentarios con ***//*** **y** ***/* */***
- Existen **palabras reservadas** del lenguaje.
- **No se define el tipo de las variables** y puede almacenar cualquier tipo de datos durante la ejecución.

- Es la principal diferencia con respecto a otros lenguajes **fuertemente tipados**.
- Se utiliza la palabra reservada **var** y el nombre de la variable solo puede contener letras, números y los símbolos "_" y "$", además de no empezar por numero.

```
var nombre="pepe";
var edad=20;
var una=23, otra='cadena'; //variables en la misma linea
var correcta; //no es necesario inicializar la variable
var 1incorrecta=2; //error no puede empezar por numero
var _sub="hola",$dolar30=30;
visible=7; /*no poner var no provoca error en el script
             pero se convirtiria en una variable global*/
```

- Otros ejemplos:

```
var numero1=45;
var numero2=12;
var resultado=numero1+numero2;
resultado='Hola';
numero1=3+4;
resultado=numero1-numero2;
resultado=resultado+4;
```

- Existen operadores aritméticos: **+,-,*,/,%, ++,--,+=,-=** ,etc
- De comparación: **==,!=,>,<,>=,<=, ===, !==** (valor y tipo)
- Lógicos: **!(NOT), &&(AND) y || (OR)**

- Aunque no es necesario declarar el tipo de las variables, es importante saber de que tipos básicos disponemos.
- Numero enteros y decimales usando el punto (10.5)
- Lógicos o booleanos (true y false).
- Texto usando comillas simple, dobles y el caracter \

```
var texto="un mensaje 'con comillas' dentro";
var texto='un mensaje "con comillas" dentro';
var texto='un mensaje \'con comillas\' dentro';
var texto="un mensaje \"con comillas\" dentro";
var texto="un mensaje "con comillas" dentro"; //no valido
var texto='un mensaje 'con comillas' dentro'; //no valido
```

Más cosas acerca del tipo texto:

- \n representa **salto de línea**.
- \t representa **tabulador**.
- + es el operador de **concatenación**.
- ¿Cuál sería el resultado en cada caso?

```
var texto1="Hola "+ "Mundo";
var texto2="Nota: "+ 10;
var texto3="7"+8;
```

- JavaScript cuenta con el tipo Array como un tipo básico.
- Un **array es una colección de variables**, que pueden ser todas del mismo tipo o cada una de un tipo diferente.
- Al contrario que la mayoría de los lenguajes.
- Los arrays **son dinámicos** por lo que pueden cambiar de **tamaño** y no hace falta indicarlo al declararlos.

```
var nombre_array=[];
var dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"];
alert(dias[0]);
alert(dias.length);
alert(dias);
```

- Aunque en JavaScript no es necesario indicar el tipo de las variables, a veces necesitamos **convertir una variable de un tipo a otro para realizar operaciones** o conocer de que tipo es el valor almacenado.
- **String(valor):**Convierte el valor indicado en los paréntesis en una cadena de texto.
- **parseInt(valor):**Convierte el valor indicado en un número entero.
- **parseFloat(valor):** Convierte el valor indicado en un número decimal.

- **typeof(valor):** Devuelve el tipo de la variable indicada. Puede ser **number, string, boolean u Object**.
- Algunos ejemplos:

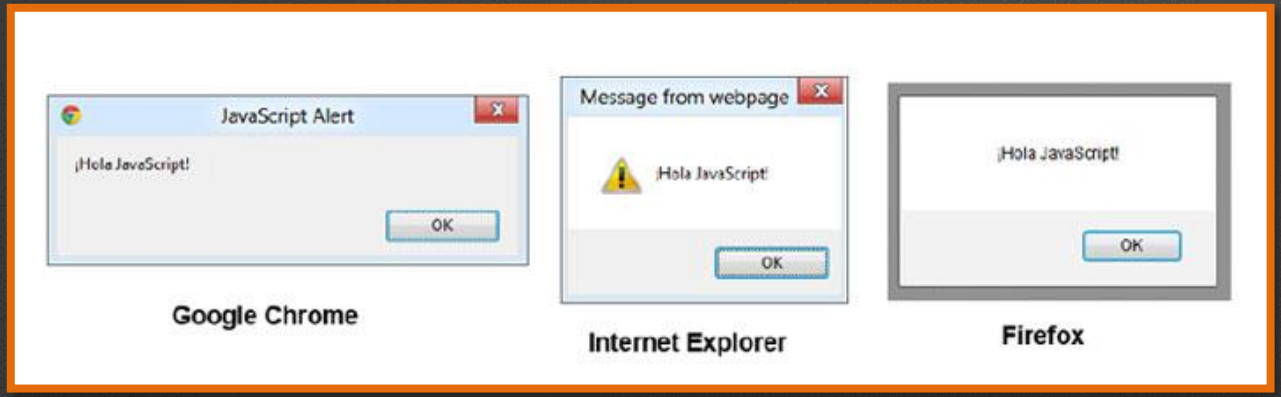
```
var num1=String(10.5); // num vale "10.5"
var num2=parseInt("2013"); // num vale 2013
var num3=parseFloat('10.5'); // num vale 10.5

var resultado;
resultado=3+"1982";
resultado=3+parseInt("1982");
resultado=num1+1;
```

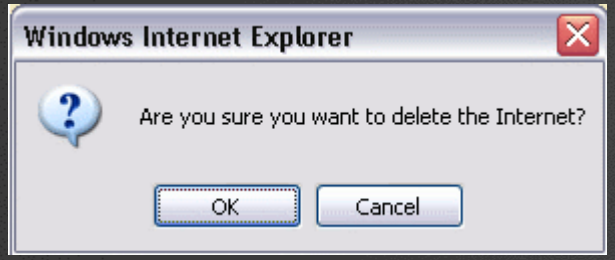
- El contenido de una variable puede ser un valor especial.
- **undefined:** valor por defecto de una variable no objeto.
- **null:** valor por defecto de una variable tipo Object
- **NaN:** Indica que una operación aritmética no devuelve un valor numérico.
- **Infinity:** Indica que se ha obtenido un valor infinito.

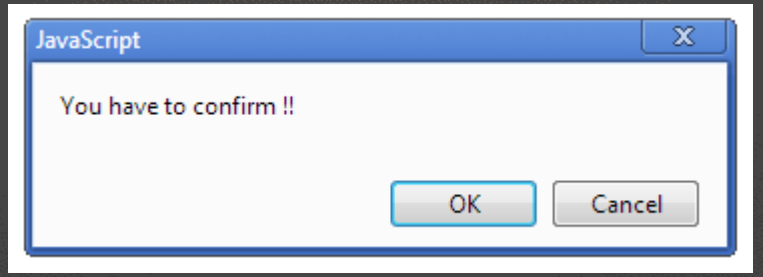
```
var num; alert(num);// por pantalla se mostrará undefined
var num={}; alert(num); //por pantalla se mostrará [object Object]
var num= 0/0; alert(num)// se mostrará NaN (Not a Number)
var num= 3/0; alert(num) //mostrará Infinity
```

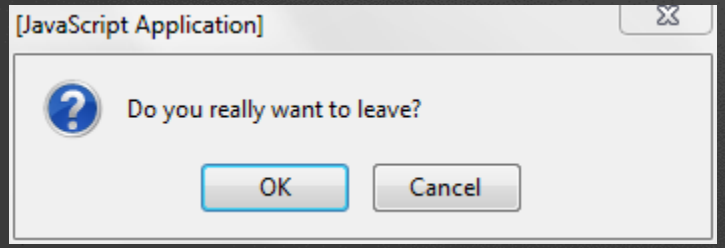
- Las funciones principales para poder **realizar interacción básica con el usuario son**:
- **alert:** La hemos utilizado anteriormente. Muestra una pantalla con el mensaje que le hayamos indicado.



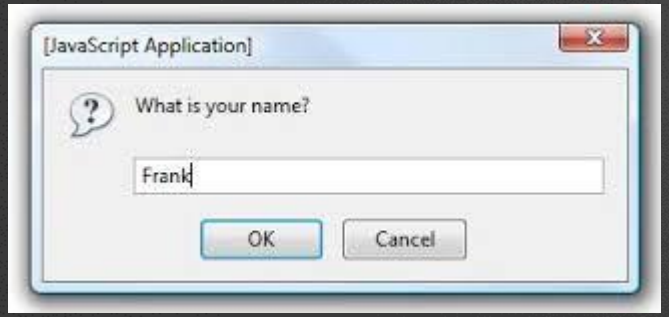
- **confirm:** Funciona igual que alert, pero además muestra los botones de aceptar y cancelar, devolviendo true o false según el botón pulsado.

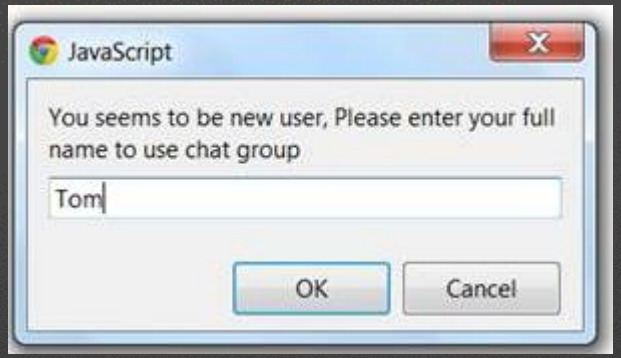






- **prompt:** Muestra el mensaje introducido entre paréntesis pero además muestra **un cuadro de texto** donde podemos escribir. Devuelve el contenido introducido si se pulsa aceptar ó null si se pulsa cancelar.





- El valor devuelto con **prompt siempre es una cadena de texto**. Si se desea trabajar con él como entero o decimal **es necesario transformarlo con parseInt o parseFloat**.

```
alert("Hola");
alert(num);
alert("Hola"+num);
var valor=confirm("Desea salir del la pagina");
var nombre=prompt("¿Como te llamas?");
var edad=prompt("¿Dime tu edad?");
edad=parseInt(edad);
```

- **document.write():** El objeto **document hace referencia a la página actual** del navegador y tiene infinidad de posibilidades, usos y métodos para trabajar sobre él.
- Por ahora nos quedaremos con **el método write que permite escribir un mensaje** en el documento actual.
- Si incluimos **etiquetas html en el mensaje**, estas **se interpretan** y se añaden al documento.

```
document.write("Hola");
document.write("<h1>Hola Mundo</h1>");
```

Fin del Tema 1

Bibliografía

- Gauchat, Juan Diego: **“El gran libro de HTML5, CSS3 y Javascript”**.Editorial Marcombo. 2012
- Vara, J.M. y otros: **“Desarrollo Web en Entorno Cliente. CFGS”**.Editorial Ra-Ma. 2012
- **“Programación en Javascript”.** Colección de artículos disponibles en la url. http://www.desarrolloweb.com/manuales/ Última visita: Septiembe 2017.
- **W3SCHOOL “Manual de referencia y Tutoriales”** http://www.w3schools.com/ Última visita Septiembre 2017
- **Comesaña, J.L. : Técnico en Desarrollo de Aplicaciones Web.** http://www.sitiolibre.com/daw.php Última visita Septiembre 2017