Introducción

En estos apuntes vamos a contar algo de Javascript. Vamos a ver sólo una parte, que es la que está en este curso y dándole un enfoque lo más práctico posible, suprimiendo toda la teoría, que la puedes encontrar en la bibliografía.

¿CÓMO SE PONE?

JavaScript es un lenguaje para que el navegador tome acciones. Se usa para que las páginas sean más interactivas, que se muevan, que respondan, que hagan cosas, sin necesidad de ir a Internet a por otra página.

En los primeros ejemplos se ha ofrecido un primer contacto en el que se ilustran algunas posibilidades iniciales.

Las instrucciones JavaScript hay que ponerlas entre las etiquetas <SCRIPT> y </SCRIPT> Puede ir en la zona <head> </head> (lo más habitual) o en la <body> </body> Por ejemplo:

```
<!DOCTYPE html">
<html>
<head>
</head>
<body>
<script language="Javascript">
alert("Adiós Mundo");
</script>
</body>
</html>
```

Al abrirla en el navegador te tiene que salir una ventanita con el mensaje.

Observa que se ha añadido una etiqueta `<!DOCTYPE ...` Algunos navegadores no la necesitan; otros se lían si no está. Por si acaso, cópiala siempre, tal y como aparece ahí.

Haz tú una página que saque una ventana con tu nombre. Es como la anterior, pero cambias lo de Adiós Mundo por tu nombre.

También se puede escribir las instrucciones Javascript en un archivo separado con la extensión .js, y posteriormente incluir entre las etiquetas <head> </head> el siguiente código:

<script language="javascript" src="archivo.js"></script>

Debes sustituir archivo.js por el nombre de tu archivo .js; este archivo debe estar en la misma carpeta que tu página web.

Aplicado al ejemplo anterior, la página quedaría:

```
<!DOCTYPE html">
<html>
<head>
</head>
<body>
<script language="Javascript" src=”programa.js”>
</script>
</body>
</html>
```

Y en “programa.js”, que debe estar en la misma carpeta, tendremos sólo:

```
alert("Adiós Mundo");
```

Si la pruebas en el navegador verás que el resultado es el mismo que antes.

Haz tú lo mismo con la que presenta tu nombre.

Javascript es sensible a mayúsculas y minúsculas. No es lo mismo escribir document.write que DOCUMENT.WRITE. Nos acostumbraremos a prestar atención cada vez que escribamos en minúsculas o mayúsculas para no cometer errores.

A veces ponemos comentarios en Javascript. Están en nuestro idioma y los ponemos para aclararnos nosotros o alguien que lo vaya a mirar. El navegador no tiene que hacer nada con ellos y eso hay que indicárselo. La forma de introducir comentarios en JavaScript es con // si se trata de una línea, o con /* ...*/, si queremos tomar un bloque como comentario.

Por ejemplo, podemos añadir al fichero "programa.js" lo siguiente:

```
// Programa que saca sólo un mensaje
```

Pon también tú algún comentario en el programa. Visualízala en el navegador. No tiene que haber ninguna diferencia.

El lenguaje Javascript no reconoce los saltos de línea, es decir, es lo mismo poner

```
var_1 =
10 ;
var_1 = 10;
```

En este curso vamos a usar una ampliación de Javascript llamada jQuery, que simplifica unas cuantas cosas. Cuando usamos jQuery, estamos en este último caso de tener un archivo JavaScript aparte (eso es en realidad jQuery), por lo que antes de nuestro Javascript pondríamos, si lo tenemos como se descarga de la página del curso:

```
<script src="jquery.js"></script>
```

Si tienes otra versión con otro nombre, cámbialo acordemente.

Variables

Una variable es un depósito donde hay un valor, y tiene un nombre, es decir, que las variables son nombres que ponemos a los lugares o memorias donde almacenamos la información. En JavaScript, deben comenzar por una letra, pudiendo haber dígitos entre los demás caracteres. Hay algunas palabras que tienen un uso especial en Javascript y no pueden usarse para variables, como `var, if, for`, etc; todas las que veamos que hacen algo particular.

Una variable se puede definir (aunque no es necesario) anteponiéndole la palabra clave `var`:

```
var dia;
```

Se pueden declarar varias variables en una misma línea:

```
var dia, mes, anio;
```

A una variable se la puede definir e inmediatamente darle un valor:

```
 var edad=20;
```

o en su defecto en dos pasos:

```
var edad;
edad=20;
```

También se puede simplemente escribir el nombre que tú quieras para la variable, luego un signo de igual (=) y finalmente el valor inicial que tendrá.

Ejemplo:

```
<script language="Javascript">
<!--
variable_1 = window.location
alert ( variable_1 )
-->
```

```
</script>
```

Como vemos en la variable_1 se guarda la información sobre la dirección de esta página, que obtenemos con window.location, posteriormente mostramos el valor de variable_1 en el mensaje de alerta que se muestra con alert( ), dentro del paréntesis va el nombre de la variable_1 y por lo tanto ese es lo que se muestra.

Debemos elegir nombres de variables representativos. Los nombres dia, mes, anio son lo suficientemente claros para darnos una idea sobre su contenido; una mala elección de nombres hubiera sido llamarlas a,b y c. Otros no son tan representativos, por ejemplo d, m, a. Posiblemente cuando estemos resolviendo un problema dicho nombre nos recuerde que almacenamos el dia, pero pasado un tiempo lo olvidaríamos.

Cuando mostramos el valor de una variable, no la ponemos entre comillas (en caso de hacer esto, aparecerá el nombre de la variable y no su contenido).

Cuando damos valores, los textos (es decir, caracteres) deben ir entre comillas. Los números (por ejemplo 10) no deben ir entre comillas.

Cada instrucción finaliza con un punto y coma.

Podemos tener variables de tipo boolean, que pueden almacenar sólo dos valores: true o false.

En JavaScript una variable se puede usar para un número y luego para un texto, por ejemplo:

```
var unValor=50
```

Y después asignar a un Valor un valor de tipo texto:

```
unValor="Ahora está lloviendo"
```

Cuando a una variable no se le asigna un valor, tiene valor indefinido (undefined). Si se intenta acceder a ella pueden ocurrir dos cosas:

-Si fue declarada sin "var", se produce un error de ejecución.

-Si fue declarada con "var", tiene el valor NaN (Not a Number).

Veamos un ejemplo:

```
x=y-2;// Si y no ha sido definida provoca un error de ejecución
var y;
x=y-2;// da el valor NaN
```

Acceso a la página HTML

Una de las tareas habituales en la programación de aplicaciones web con JavaScript consiste en la manipulación de las páginas web. Por ejemplo, es habitual obtener el valor almacenado por algunos elementos (por ejemplo los elementos de un formulario), crear un elemento (párrafos, , etc.) de forma dinámica y añadirlo a la página, aplicar una animación a un elemento (que aparezca/desaparezca, que se desplace, etc.).

Acceso a los elementos de la página

El acceso a los elementos de la página, su modificación y su eliminación, solamente es posible después de que la página se cargue por completo.

Formas de acceder a elementos con jQuery

Si sabemos que su ID es “elemento1”

```
$("#elemento1")
```

Si queremos un grupo con todos los elementos del tipo “tipo” (puede ser p para párrafos, a para enlaces, etc. según las etiquetas HTML)

```
$("tipo")
```

Podemos operar directamente sobre todo el grupo; por ejemplo, poner todos los enlaces apuntando a la página de la universidad.

Seleccionar un grupo según una clase CSS; por ejemplo, los de clase “invisible”:

```
$(".invisible")
```

Se pueden combinar criterios; por ejemplo el elemento de ID “elemento1” y todos los de la clase CSS “invisible”:

```
$("#elemento1, .invisible")
```

Se pueden indicar que estén dentro de otros; por ejemplo, todos los párrafos dentro de “elemento1”

```
$(“#elemento1 > p”)
```

Los elementos seleccionados se pueden guardar en una variable. Si queremos comprobar cuántos son lo podemos hacer con la propiedad `.length` Por ejemplo, si quiero guardar los elementos de la clase *base* en la variable `elems` y cuántos son en la variable `cuanel`, haría:

```
elems=$(“.base”)
cuanel=elems.length
```

Añadido de contenido

Con jQuery se puede cambiar el contenido HTML del elemento. Por ejemplo, para poner en “elemento1” un párrafo que diga “Listo”

```
 $("#elemento1").html("<p>Listo</p>");
```

Si lo que queremos es añadirlo al final de lo que haya, sería:

```
$("#elemento1").append("<p>Listo</p>");
```

Si queremos añadirlo al principio, en vez de append, usaríamos prepend.

Eliminación

Con jQuery, se puede reemplazar un elemento por otra cosa. Por ejemplo, para cambiar “elemento1” por un párrafo que diga “Fase 2”:

```
   $("#elemento1).replaceWith("<p>Fase 2</p>");
```

Y también se puede directamente borrarlo:

```
    $("#elemento1").remove();
```

Acceso a los atributos

Se pueden modificar los atributos y propiedades de los elementos. Es posible acceder de forma sencilla a todos los atributos y todas las propiedades CSS de cualquier elemento de la página.

Algunas de las propiedades y funciones que nos pueden interesar habitualmente son: height, innerHTML, length, offsetHeight, offsetLeft, offsetTop, offsetWidth, width, onclick, ondblclick, onmouseover y en general todos los atributos que se han comentado en HTML.

El único atributo que cambia de nombre es class. Como la palabra class tiene un significado especial en JavaScript, no es posible utilizarla para acceder al atributo class. En su lugar se utiliza el nombre className para acceder al atributo class

Operación con Jquery

Para cambiar atributos añadimos .attr y entre paréntesis, la lista de atributos a cambiar y sus valores. Ejemplo: poner todas las imágenes de la página a “/images/hat.gif”

```
    $("img").attr({
          src: "/images/hat.gif"
         });
```

Si hay más pares atributo/valor, se separan por comas.

Para establecer propiedades CSS de un elemento , se usa css. Por ejemplo, para poner todos los párrafos en rojo y negrita con el fondo azul:

```
$("p").css({ color: "red", background: "blue", “font-weight”: "bold" });
```

Observa que si la propiedad lleva un guión va entre comillas. También puede ir sin comillas si quitamos el guión y ponemos la siguiente letra a mayúscula (fontWeight).

Añadiendo al objeto la sintaxis .show o .hide y entre paréntesis “slow” o”fast”, se consigue un efecto de aparición o desaparición más o menos progresiva, según indiquemos. Por ejemplo, si en parrafo tenemos una referencia a un párrafo que queremos hacer desaparecer: `parrafo.hide("slow");`

Si lo que queremos es simplemente conocer una propiedad, usaremos `attr` o `css` según corresponda, con el nombre de la propiedad entre comillas y paréntesis. Por ejemplo, si quiero meter en la variable `fich` el nombre de fichero de la imagen cuyo `id` es *principal* y en la variable `cualcolor` el color que tiene la letra del párrafo cuyo `id` es *primero*, haría:

```
fich=$(“#principal”).attr(“src”);
cualcolor=$(“#primero”).css(“color”);
```

Leer datos de teclado

Para la entrada de datos por teclado tenemos la función prompt. Cada vez que necesitamos ingresar un dato con esta función, aparece una ventana donde cargamos el valor. Se pueden usar también formularios, pero a veces nos resultará más práctica esta función. Para ver su funcionamiento analicemos este ejemplo:

```
<html>
  <head>
  </head>
  <body>
  <script language="JavaScript">
  var nombre;
  var edad;
  nombre=prompt('Tu nombre:','');
  edad=prompt('Tu edad:','');
  </script>
  </body>
  </html>
```

La forma de usar la función prompt es:

```
<variable que recibe el dato>=prompt(<mensaje a mostrar en la ventana>,<valor
  inicial a mostrar en la ventana>);
```

La función prompt tiene dos parámetros: uno es el mensaje y el otro el valor inicial a mostrar.

Cálculos directos

Ejemplo: cargar dos números por teclado e imprimir su suma y su producto:

```
<html>
  <head>
  <script language="JavaScript">
  var valor1;
  var valor2;
  valor1=prompt('Ingrese primer número:','');
  valor2=prompt('Ingrese segundo número','');
  n1=parseInt(valor1)
  n2=parseInt(valor2)
  var suma=n1+n2;
  var producto=n1*n2;
  alert('La suma es '+suma+' El producto es '+producto);
  </script>
  </head>
  <body>
  </body>
  </html>
```

Lo primero que debemos tener en cuenta es que si queremos que el operador + sume los contenidos de los valores numéricos ingresados por teclado, debemos llamar a la función parseInt y pasarle como parámetro las variables valor1 y valor2 sucesivamente. Con esto logramos que el operador más, sume las variables como enteros y no como textos. Si por ejemplo sumamos 1 + 1 sin utilizar la función parseInt el resultado será 11 en lugar de 2, ya que el operador + concatena los dos textos.

Una vez que ya los tenemos pasados a números, podemos hacer todas las operaciones numéricas que queramos.

Recordemos que hay que tener cuidado cuando operamos, no vaya a ser que confundamos los contenidos entre textos y números.

Existen varios tipos de operadores en JavaScript:

Asignación

Este tipo de operador se utiliza para dar valores a las variables.

```
var resultado=50
```

Da a la variable "resultado" el valor 50.

Otro ejemplo:

```
X = 3
Y = 2
X = Y
```

al final X vale 2

Aritméticos

Los operadores aritméticos, a partir de varios operandos devuelven un solo valor; resultado de la operación realizada con los anteriores operandos.

A continuación se resumen los más comunes.

+ Suma dos números o junta dos textos.
  X = 5 ; Y = 2 ; X + Y da 7
  Z = "Ho"; W = "la" ; Z + W da "Hola"

- Substrae la segunda variable a la primera.
  X = 4; Y = 1; X - Y da 3

* Multiplica dos números
  X = 2 ; Y = 2; X * Y vale 4

/ Divide dos numéros
X = 6; Y = 2; X / Y vale 3

Además está el cambio de signo:

- : cambia el signo de una variable.

Fecha y hora

Existe un conjunto útil de funciones para fecha y hora, veamos algunas de ellas.

Para sacar el momento actual tenemos:

fecha = new Date();

En vez de fecha, podemos poner cualquier variable. A partir de ella tenemos

fecha.getDate(): día del mes

fecha.getDay(): día de la semana (numéricamente: 1 para lunes, etc.)

fecha.getMonth(): número del mes

fecha.getYear(): año (contado a partir de 1900: el 2014 dará 114)

fecha.getFullYear(): año (completo)

fecha.getTime(): milisegundos transcurridos desde 1/1/1970

fecha.getHours(): hora entre 0 y 23

fecha.getMinutes(): minutos entre 0 y 59

fecha.getSeconds(): segundos entre 0 y 59

Por ejemplo, para meter el día del mes en la variable diames sería:

diames=fecha.getDate()