**Programador Web Inicial**

**Javascript ES 6**

Presentación:

En esta unidad veremos la sintaxis de javascript, como trabaja con el navegador web y haremos nuestro primer pequeño código.

Qué son los eventos, como capturar y trabajar con estos y como validar formularios.

Algunas particularidades de ECMAscript 6

Bloques temáticos:

* Sintaxis
* Variables
* Operadores
* Estructuras
* Funciones – Arrays
* Formularios
* Campos de texto
* Checkbox
* Select
* Eventos
* Seleccionar elemento e imprimir en html
* Definir una constante
* Let
* Función Arrow
* Clases
* Template Strings
* Valores por defecto
* Módulos

# Javascript – Sintaxis

## Comentarios

Text

Description automatically generated with low confidence

En caso de utilizar archivos JS minificados, se deben utilizar comentarios extendidos y NO de una línea.

\* Archivo minificado -> un js minificado es “comprimir” todo el contenido de nuestro js en un archivo de una única línea.

## Separación de instrucciones

Javascript tiene dos maneras de separar instrucciones:

* La primera es a través del carácter punto y coma (;)



* La segunda es a través de un salto de línea.



# Javascript – Variables

## Variables

**Una variable es un espacio en memoria donde se almacena un dato**, un espacio donde podemos guardar cualquier tipo de información que necesitemos para realizar las acciones de nuestros programas.

Los nombres de las variables han de construirse con caracteres alfanuméricos y el carácter subrayado (\_).

Los nombres tienen que comenzar por un carácter alfabético o el subrayado. **No podemos utilizar caracteres raros como el signo +, un espacio.**

**Las variables no pueden utilizar nombres reservados, por ejemplo if, for, while, etc**

## Declaración de Variables

Declarar variables consiste en definir y de paso informar al sistema de qué vas a utilizar una variable.

Javascript cuenta con la palabra "**var**" que utilizaremos cuando queramos declarar una o varias variables. Como es lógico, se utiliza esa palabra para definir la variable antes de utilizarla.

Ejemplo:

Graphical user interface

Description automatically generated with medium confidence

Ejemplo:

Text, letter

Description automatically generated

Nota: Si nosotros no declaramos el **modo estricto**, podremos utilizar variables sin necesidad de declararlas con la palabra reservada **var.**

Para declarar el **strict mode** debemos anteponer al código **“strict mode”** por ejemplo:

Graphical user interface, text

Description automatically generated

En este caso la primer asignación se hace sin modo estricto por lo cual como vemos no tiene la palabra var adelante.

En cambio dentro de la función f1 se hace en modo estricto por lo cual debe tener la palabra var delante.

## Ámbito de las Variables

**Se le llama ámbito de las variables al lugar donde estas están disponibles.** Por lo general, cuando declaramos una variable hacemos que esté disponible en el lugar donde se ha declarado.

* Variables globales: son las que están declaradas en el ámbito más amplio posible.

Text

Description automatically generated

* Variables locales: sólo podremos acceder a ellas dentro del lugar donde se ha declarado

Graphical user interface, text, application

Description automatically generated

# Javascript – Operadores

## Operadores de cadenas

Las cadenas de caracteres, o variables de texto, también tienen sus propios operadores para realizar acciones típicas sobre cadenas. Aunque javascript sólo tiene un operador para cadenas se pueden realizar otras acciones con una serie de funciones predefinidas en el lenguaje que veremos más adelante.

Para concatenar 2 cadenas de texto debemos usar el operador “**+**”

Text

Description automatically generated with low confidence

## Operadores lógicos

Estos operadores sirven para realizar operaciones lógicas, que son aquellas que dan como resultado un verdadero o un falso, y se utilizan para tomar decisiones en nuestros scripts.

Graphical user interface, text

Description automatically generated with medium confidence

## Typeof

Para comprobar el tipo de un dato se puede utilizar otro operador que está disponible a partir de javascript 1.1, el **operador typeof**, que devuelve una cadena de texto que describe el tipo del operador que estamos comprobando.

Graphical user interface, text

Description automatically generated

## Operadores de asignación

Un operador de asignación asigna un valor al operando de la izquierda en función del valor del operando de la derecha. El operador básico de asignación es el de igual (=), que asigna el valor del operando de la derecha al operando de la izquierda. Por ejemplo, x = y, está asignando el valor de y a x.

Table

Description automatically generated

## Operadores de comparación

Un operador de comparación compara sus operandos y devuelve un valor lógico en función de si la comparación es verdadera (true) o falsa (false). Los operadores pueden ser numéricos, de cadena de caracteres (Strings), lógicos o de objetos.

Graphical user interface, table

Description automatically generated with medium confidence

## Operadores aritméticos

Los operadores aritméticos toman los valores numéricos (tanto literales como variables) de sus operandos y devuelven un único resultado numérico. Los operadores aritméticos estándar son la suma (+), la resta (-), la multiplicación (\*) y la división (/).

Graphical user interface, text, application, email

Description automatically generated

# Estructuras

## IF

La estructura if nos permitirá mediante el uso de operadores de comparación tomar una decisión a partir de si dicha comparación o valor de una expresión es verdadera o falsa (en este caso entra por el else)

Graphical user interface, text, application

Description automatically generated

## For

La estructura for nos permitirá recorrer un array en javascript.

Cada elemento recorrido se reconoce como una iteración, se “cortara” el for cuando la condición sea falsa.

Graphical user interface

Description automatically generated with medium confidence

* Inicialización: Aquí inicializamos las variables. Por ejemplo: **i=0**
* Condición: Mientras la condición sea verdadera se seguirán produciendo iteraciones. Ejemplo **i<5**
* Actualización: se actualiza la variable de la condición. Ejemplo: **i++**

## While

La estructura while es similar a la for, nos permite realizar iteraciones sobre un array

Graphical user interface, text, application

Description automatically generated

Como vemos no cuenta con el elemento de inicialización ni el de actualización.

# Funciones - Arrays

## Definición

Primero se escribe la palabra **function**, reservada para este uso. Seguidamente se escribe el nombre de la función, que como los nombres de variables puede tener números, letras y algún carácter adicional como en guión bajo.

Graphical user interface, application

Description automatically generated

Entre paréntesis, luego del nombre de la función, se definirán los parámetros que recibirá la misma. En el caso de js no hace falta el tipo de dato en cada parámetro recibido.

## Arrays

El primer paso para utilizar un array es crearlo. Para ello utilizamos un objeto Javascript ya implementado en el navegador.

Graphical user interface, text, application, email

Description automatically generated

A diferencia de PHP no podemos tener vectores asociativos, es decir arrays cuya clave sea un string.

## Longitud de un array

Para obtener la longitud de un array utilizaremos el método **length**

A picture containing company name

Description automatically generated

# Formularios

Acceder a formularios con document



Acceder al formulario a través del objeto document y el nombre del formulario.



Acceder al formulario a través del objeto document y el índice del formulario.

Acceder a campos dentro del formulario



Acceder al valor del campo



Evento onclick

Con este evento definimos una acción (función) a ejecutar cuando se hace click en un elemento (un botón)



## Propiedades

* **Action**: Es la acción que queremos realizar cuando se submite un formulario. ● **Encoding**: El tipo de codificación del formulario ● **Length**: El número de campos del formulario.
* **Method**: El método por el que mandamos la información. Corresponde con el atributo METHOD del formulario.
* **Name**: El nombre del formulario, que corresponde con el atributo NAME del formulario.
* **Target**: La ventana o frame en la que está dirigido el formulario. Cuando se submite se actualizará la ventana o frame indicado. Corresponde con el atributo target del formulario.

Ejemplo de propiedad method:



## Métodos

* **submit():** Para hacer que el formulario se submite, aunque no se haya pulsado el botón de submit.
* **reset():** Para reiniciar todos los campos del formulario, como si se hubiese pulsado el botón de reset.

# Campos de texto

## Propiedades

* **defaultValue**: Es el valor por defecto que tiene un campo. Lo que contiene el atributo VALUE de la etiqueta <INPUT>.
* **Form**: Hace referencia al formulario.
* **Name**: Contiene el nombre de este campo de formulario ● **Type**: Contiene el tipo de campo de formulario que es.
* **Value**: El texto que hay escrito en el campo.

## Métodos

* **blur():** Retira el foco de la aplicación del campo de texto.
* **focus():** Pone el foco de la aplicación en el campo de texto.
* **select():** Selecciona el texto del campo.

## Javascript – Checkbox

### Propiedades

* **checked**: Informa sobre el estado del checkbox. Puede ser true o false.
* **defaultChecked**: Si está chequeada por defecto o no.
* **Value**: El valor actual del checkbox.

### Métodos

* **click():** Es como si hiciésemos un click sobre el checkbox, es decir, cambia el estado del checkbox.
* **blur():** Retira el foco de la aplicación del checkbox.
* **focus():** Coloca el foco de la aplicación en el checkbox.

## Javascript – Select

### Propiedades

* **length**: Guarda la cantidad de opciones del campo select. Cantidad de etiquetas <OPTION>
* **Option**: Hace referencia a cada una de sus opciones. Son por sí mismas objetos.
* **Options**: Un array con cada una de las opciones del select.
* **selectedIndex**: Es el índice de la opción que se encuentra seleccionada.

### Métodos

* **blur():** Para retirar el foco de la aplicación de ese elemento de formulario.
* **focus():** Para poner el foco de la aplicación.
* **Objeto option:**
  + **defaultSelected**

Indica con un true o un false si esa opción es la opción por defecto.

* + **index**

El índice de esa opción dentro del select.

* + **selected**

Indica si esa opción se encuentra seleccionada o no.

* + **text**

Es el texto de la opción. Lo que puede ver el usuario en el select, que se escribe después de la etiqueta <OPTION>.

* + **value**

Indica el valor de la opción, que se introduce con el atributo VALUE de la etiqueta <OPTION>.

# Eventos

## Manejadores

* **onblur**

Se desata un evento onblur cuando un elemento pierde el foco de la aplicación. El foco de la aplicación es el lugar donde está situado el cursor, por ejemplo puede estar situado sobre un campo de texto, una página, un botón o cualquier otro elemento.

* **onchange**

Se desata este evento cuando cambia el estado de un elemento de formulario, en ocasiones no se produce hasta que el usuario retira el foco de la aplicación del elemento. ● **onclick**

Se produce cuando se da una pulsación o clic al botón del ratón sobre un elemento de la página, generalmente un botón o un enlace.

* **Onload**

Este evento se desata cuando la página, o en Javascript 1.1 las imágenes, ha terminado de cargarse.

* **onsubmit**

Ocurre cuando el visitante apreta sobre el botón de enviar el formulario. Se ejecuta antes del envío propiamente dicho.

Para ver más manejadores: [***http://www.desarrolloweb.com/articulos/1236.php***](http://www.desarrolloweb.com/articulos/1236.php) Ejemplo de on blur:

A picture containing timeline

Description automatically generated

## Javascript – Seleccionar elemento e imprimir en html

### getElementById

Permite, como su nombre indica, seleccionar un elemento del documento por medio del valor del atributo id que se le haya asignado

A picture containing text

Description automatically generated

### InnerHTML

Retorna o setea el contenido HTML de un elemento

* Set



* Get



## Javascript – Definir una constante

En javascript ES 6 podemos definir una constante con la palabra reservada **const**



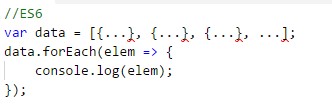
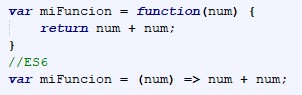
## Javascript – Let

Podemos utilizar **let** en lugar de **var** a la hora de declarar una variable, cuando queremos que esta solo sea accedida de manera local en determinado ámbito. Por ejemplo:

Graphical user interface, text

Description automatically generated

## Javascript – Función Arrow



En ambos ejemplos se puede que se sustituye el uso de la palabra reservada **function** dando simplicidad en el código.

**(parametro1,parametro2,…,parámetro n) =>{Definición de la función}**

## Javascript – Clases

Ahora JavaScript tendrá clases, muy parecidas las funciones constructoras de objetos que realizamos en el estándar anterior, pero ahora bajo el paradigma de clases, con todo lo que eso conlleva, como por ejemplo, herencia.

Text

Description automatically generated

**This**

La variable this muchas veces se vuelve un dolor de cabeza. antiguamente teníamos que cachearlo en otra variable ya que solo hace referencia al contexto en el que nos encontremos. Por ejemplo, en el siguiente código si no hacemos var that = this dentro de la función document.addEventListener, this haría referencia a la función que pasamos por Callback y no podríamos llamar a foo()

Graphical user interface, text, application

Description automatically generated

Con ECMAScript5 la cosa cambió un poco, y gracias al método bind podíamos indicarle que this hace referencia a un contexto y no a otro.

Graphical user interface, text, application

Description automatically generated

Ahora con ES6 y la función Arrow => la cosa es todavía más visual y sencilla.

Graphical user interface, text

Description automatically generated

## Javascript – Template Strings

Con ES6 podemos interpolar Strings de una forma más sencilla que como estábamos haciendo hasta ahora. Fíjate en este ejemplo:

Graphical user interface, text, application

Description automatically generated

También podemos tener String multilínea sin necesidad de concatenarlos con +

A picture containing graphical user interface

Description automatically generated

## Javascript – Valores por defecto

Otra novedad es asignar valores por defecto a las variables que se pasan por parámetros en las funciones. Antes teníamos que comprobar si la variable ya tenía un valor. Ahora con ES6 se la podemos asignar según creemos la función.

Text

Description automatically generated

## Javascript – Módulos

Ahora JavaScript se empieza a parecer a lenguajes como Python o Ruby. Llamamos a las funciones desde los propios Scripts, sin tener que importarlos en el HTML, si usamos JavaScript en el navegador.

Graphical user interface, text, application

Description automatically generated

Y para importar en otro fichero:

Text

Description automatically generated

Bibliografía y Webgrafía utilizada y sugerida

Fedosejev, A. (2015). React.js Essentials (1 ed.). EEUU, Packt.

Amler, . (2016). ReactJS by Example (1 ed.). EEUU, Packt.

Stein, J. (2016). ReactJS Cookbook (1 ed.). EEUU, Packt. [***https://www.tutorialspoint.com/es6/es6\_syntax.htm***](https://www.tutorialspoint.com/es6/es6_syntax.htm)

***<http://www.desarrolloweb.com/manuales/20/>***[***http://www.desarrolloweb.com/articulos/826.php***](http://www.desarrolloweb.com/articulos/826.php) [***http://www.desarrolloweb.com/articulos/827.php***](http://www.desarrolloweb.com/articulos/827.php) [***http://www.desarrolloweb.com/articulos/846.php***](http://www.desarrolloweb.com/articulos/846.php) [***http://www.desarrolloweb.com/articulos/861.php***](http://www.desarrolloweb.com/articulos/861.php)

[***https://carlosazaustre.es/ecmascript-6-el-nuevo-estandar-de-javascript/***](https://carlosazaustre.es/ecmascript-6-el-nuevo-estandar-de-javascript/)