**TID91D**



**DESARROLLO MÓVIL MULTIPLATAFORMA**

Reporte de investigación de la sintaxis del lenguaje Java Script y aplicación

ERIKA ELÍ DOMÍNGUEZ CARRILLO 1118150132

**MSI. Alfonso José Barroso Barajas**

Índice

[INTRODUCCION 1](#_Toc74603343)

[JAVASCRIPT 2](#_Toc74603344)

[Conceptos básicos 2](#_Toc74603345)

[Declaraciones 3](#_Toc74603346)

[Variables 3](#_Toc74603347)

[Funciones 4](#_Toc74603348)

[Declaración de función 4](#_Toc74603349)

[TYPESCRIPT 5](#_Toc74603350)

[Referencias 6](#_Toc74603351)

# INTRODUCCION

La sintaxis tiene que ver con el orden de los términos e instrucciones que permiten la correcta ejecución de un programa informático, cuando son alimentados al sistema. En ese sentido, los errores de la sintaxis también se producen y también sabotean la comprensión del mensaje o, en este caso, de las instrucciones, impidiendo así que se lleven a cabo y que el software se ejecute como debería.

Es por eso que a continuación, te mostraremos la manera correcta de desarrollar tu código en Javascript y Typescript.

# JAVASCRIPT

## Conceptos básicos

JavaScript está influenciado sobre todo por la sintaxis de Java, C y C++, pero también ha sido influenciado por Awk, Perl y Python.

JavaScript distingue entre mayúsculas y minúsculas (es case-sensitive) y utiliza el conjunto de caracteres Unicode. Por ejemplo, la palabra «Früh» (que significa "temprano" en Alemán) se podría usar como el nombre de una variable.

**let Früh = "foobar"**

Pero, la variable **früh** no es la misma que **Früh** porque JavaScript distingue entre mayúsculas y minúsculas.

En JavaScript, las instrucciones se denominan declaraciones y están separadas por punto y coma (;).

No es necesario un punto y coma después de una declaración si está escrita en su propia línea. Pero si se deseas más de una declaración en una línea, entonces debes separarlas con punto y coma.

Sin embargo, se considera una buena práctica escribir siempre un punto y coma después de una declaración, incluso cuando no sea estrictamente necesario. Esta práctica reduce las posibilidades de que se introduzcan errores en el código.

El texto fuente del script JavaScript se escanea de izquierda a derecha y se convierte en una secuencia de elementos de entrada que son fragmentos, caracteres de control, terminadores de línea, comentarios o espacios en blanco. (Los espacios, tabulaciones y caracteres de nueva línea se consideran espacios en blanco).

## Declaraciones

JavaScript tiene tres tipos de declaraciones de variables.

**var**

Declara una variable, opcionalmente la inicia a un valor.

**let**

Declara una variable local con ámbito de bloque, opcionalmente la inicia a un valor.

**const**

Declara un nombre de constante de solo lectura y ámbito de bloque.

### [Variables](https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Grammar_and_types#variables)

Utiliza variables como nombres simbólicos para valores en tu aplicación. Los nombres de las variables, llamados identificadores, se ajustan a ciertas reglas.

Un identificador de JavaScript debe comenzar con una letra, un guión bajo (\_) o un signo de dólar ($). Los siguientes caracteres también pueden ser dígitos (0-9).

Dado que JavaScript distingue entre mayúsculas y minúsculas, las letras incluyen los caracteres "A" a "Z" (mayúsculas), así como "a" a "z" (minúsculas).

Puedes utilizar la mayoría de las letras ISO 8859-1 o Unicode como å y ü en los identificadores

## Funciones

### Declaración de función

Una definición de función (también denominada declaración de función o expresión de función) consta de la palabra clave function, seguida de:

* El nombre de la función.
* Una lista de parámetros de la función, entre paréntesis y separados por comas.
* Las declaraciones de JavaScript que definen la función, encerradas entre llaves, { ... }.

Por ejemplo, el siguiente código define una función simple llamada square ("cuadrado"):

**function square(number) {**

**return number \* number;**

**}**

La función *square* toma un parámetro, llamado *number*. La función consta de una declaración que dice devuelva el parámetro de la función (es decir, *number*) multiplicado por sí mismo. La instrucción *return* especifica el valor devuelto por la función.

# TYPESCRIPT

Typescript es un lenguaje que creó Microsoft. Su sintaxis es una especie de mezcla entre Javascript y Java. Todo el código que escribas en Typescript será compilado en código Javascript.

Typescript está pensado para añadir funcionalidad a Javascript, extenderlo, y por tanto puedes usar código Javascript dentro de métodos y clases de Typescript.

Typescript añade tipado estático a Javascript, puedes crear variables con un tipo fijado (string, números, etc). El tipado estático es opcional por lo que deja al desarrollador si quiere utilizarlos.

Los tipos que vienen con typescript son:

* boolean: (true/false).
* number: integers, floats, Infinito y not a number.
* string: caractéres o lista de caracteres.
* []: Arrays of other types, like number[] or boolean[].
* {}: Objeto literal.
* undefined.
* any: Tipo especial que permite que una variable pueda tener cualquier tipo.

let **decimal: number = 6;**

También puedes declarar los tipos para los parámetros de entrada de las funciones, incluso para el tipo de dato que devuelven:

**function identity(arg: number): number {**

**return arg;**

**}**

# Referencias

*Angular - Cómo es la sintaxis de Typescript*. (Abril de 2020). Obtenido de Coding Potions: https://codingpotions.com/angular-controladores-typescript

*MDN Web Docs*. (13 de Junio de 2021). Obtenido de Mozilla: https://developer.mozilla.org/es/docs/Web/JavaScript/Guide

Raffino, M. E. (6 de Septiembre de 2020). *Sintaxis*. Obtenido de Concepto.de: https://concepto.de/sintaxis/