

JSDoc

Documentación



INDEX

1. Proyecto de JavaScript	3
a. Código JavaScript	3
b. Comentarios al código	3
2. Creación del JSDoc	5
3. Explicación del JSDoc	5

1. Proyecto de JavaScript

a. Código JavaScript

Para este proyecto de JavaScript he creado 3 archivos .java para poder generar un el JSDoc y realizar la práctica:

- Clase Persona
- Clase Estudiante
- Clase Main

Link al código:

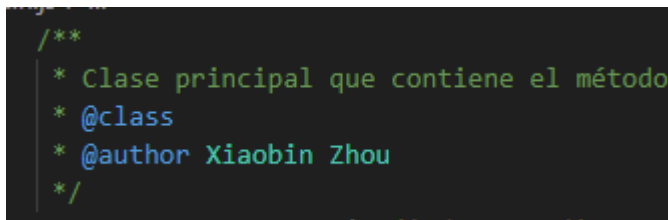
<https://github.com/xzhou12/m08-despl-app-web/tree/UF1/UF1/Practica%204%20Documentación>

b. Comentarios al código

Para generar el JavaDoc con los diferentes parámetros/etiquetas tendremos que añadir comentarios en la parte superior de cada función y cada clase:

Las diferentes etiquetas que podremos usar son las siguientes:

- **@author** : Nombre del desarrollador.

A screenshot of a code editor showing a Java class definition. The code is enclosed in a multi-line comment block (/** ... */) and includes the following JSDoc tags: @class, @author Xiaobin Zhou, and @param. The text is color-coded: green for the comment delimiters and the @class tag, and blue for the @author and @param tags.

```
/**
 * Clase principal que contiene el método
 * @class
 * @author Xiaobin Zhou
 */
```

- **@version** : Versión del método o clase.
- **@constructor** : Indica el constructor.
- **@param** : Definición de un parámetro de un método, es requerido para todos los parámetros del método.

```

/**
 * Constructor de la clase Persona.
 * @constructor
 * @param {string} nombre - El nombre de la persona.
 * @param {number} edad - La edad de la persona.
 */
constructor(nombre, edad) {
    this.nombre = nombre;
    this.edad = edad;
}

```

- **@return** : Informa de lo que devuelve el método, no se puede usar en constructores o métodos "void".

```

/**
 * Obtiene el nombre de la persona.
 * @returns {string} El nombre de la persona.
 */
getNombre() {
    return this.nombre;
}

```

- **@throws** : Excepción lanzada por el método, posee un sinónimo de nombre **@exception**.
- **@exception** : sinónimo de **@throws**.

```

/**
 * Revisa si la edad de la persona es válida.
 * @throws {Error} Si la edad es un número negativo.
 */
revisarEdad() {
    if (this.edad < 0) {
        throw new Error("La edad no puede ser un número negativo.");
    } else {
        console.log("La edad es válida.");
    }
}

```

- **@see** : Asocia con otro método o clase.
- **@serial** : Describe el significado del campo y sus valores aceptables. Otras formas válidas son **@serialField** y **@serialData**.
- **@deprecated** : Indica que el método o clase es antigua y que no se recomienda su uso porque posiblemente desaparecerá en versiones posteriores.
- **@private** : Indica que el método es privado.

2. Creación del JSDoc

Una vez que tenemos creado el proyecto de JavaScript, tendremos que instalar una herramienta para poder generar el JSDoc:

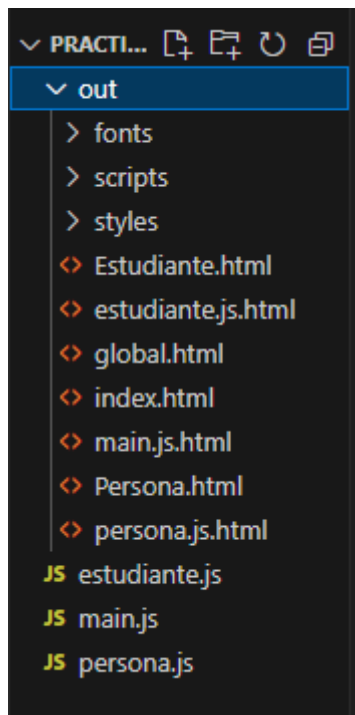
```
npm install -g jsdoc
```

Una vez instalado el “jsdoc” podremos empezar a generar el JSDoc, para ello escribiremos el siguiente comando dentro de la consola:

```
jsdoc (archivos)
```

```
Doc>jsdoc main.js estudiante.js persona.js
```

Una vez ejecutado, nos generará una carpeta llamada “out”



Donde se encuentran todos los archivos del JSDoc.

3. Explicación del JSDoc

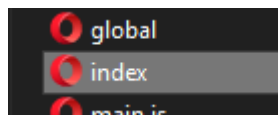
Link del JavaDoc:

<https://github.com/xzhou12/m08-despl-app-web/tree/UF1/UF1/Practica%204%20Documentación>

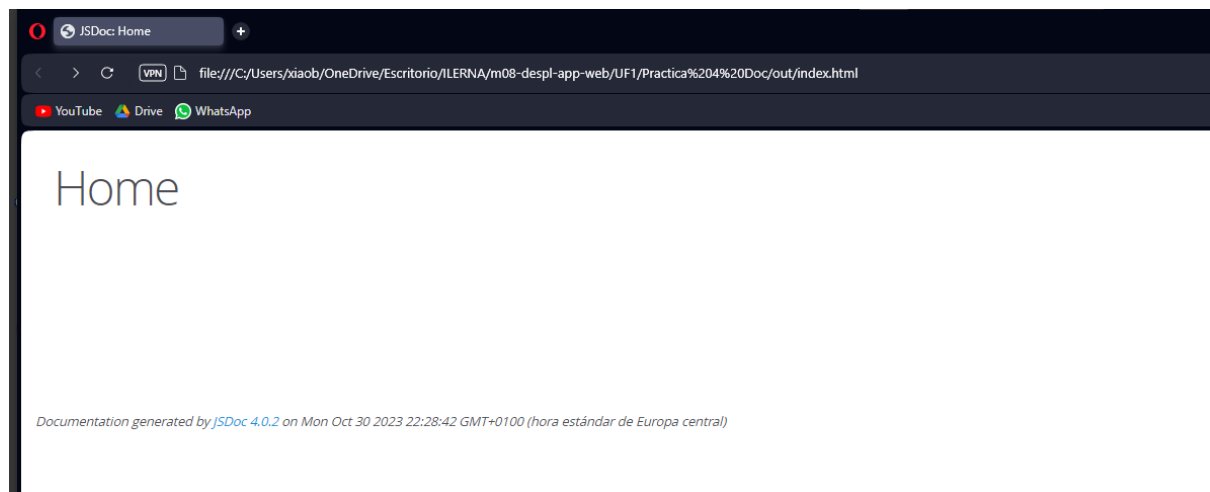
Una vez creado el JSDoc, nos creará estos diferentes archivos:

<< m08-despl-app-web > UF1 > Practica 4 Doc > out > <input type="text" value="Buscar en out"/>						
	Nombre	Estado	Fecha de modificación	Tipo	Tamaño	
	fonts	✓	30/10/2023 22:28	Carpeta de archivos		
	scripts	✓	30/10/2023 22:28	Carpeta de archivos		
	styles	✓	30/10/2023 22:28	Carpeta de archivos		
	Estudiante	✓	30/10/2023 22:28	Opera Web Docu...	5 KB	
	estudiante.js	✓	30/10/2023 22:28	Opera Web Docu...	3 KB	
	global	✓	30/10/2023 22:28	Opera Web Docu...	3 KB	
	index	✓	30/10/2023 22:28	Opera Web Docu...	2 KB	
	main.js	✓	30/10/2023 22:28	Opera Web Docu...	3 KB	
	Persona	✓	30/10/2023 22:28	Opera Web Docu...	26 KB	
	persona.js	✓	30/10/2023 22:28	Opera Web Docu...	4 KB	

Para poder visualizar el JSDoc, tendremos que entrar en el “index.html”



Y nos aparecerá este diseño:



Con el JSDoc, nos enseñará cómo está estructurado el proyecto de JavaScript, nos mostrará información sobre las funciones, las variables, los parámetros, los returns, ...

También nos muestra la estructura que sigue los objetos en el proyecto:

JSDoc: Class: Estudiante

file:///C:/Users/xiaob/OneDrive/Escritorio/ILERNA/m08-despl-app-web/UF1/Practica%204%20Doc/out/Estudiante.html

YouTube Drive WhatsApp

Class: Estudiante

Estudiante(nombre, edad, universidad)

new Estudiante(nombre, edad, universidad)

Constructor de la clase Estudiante.

Parameters:

Name	Type	Description
nombre	string	El nombre del estudiante.
edad	number	La edad del estudiante.
universidad	string	La universidad a la que asiste el estudiante.

Source: [estudiante.js, line 16](#)

Methods

estudiar()

Permite al estudiante indicar que está estudiando.

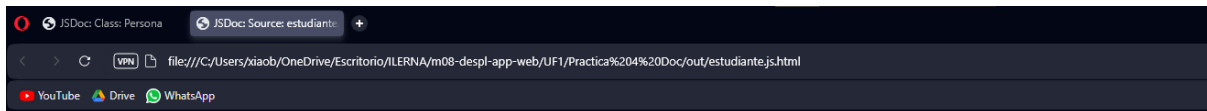
Source: [estudiante.js, line 24](#)

realizarExamen()

Permite al estudiante indicar que está realizando un examen.

Source: [estudiante.js, line 31](#)

También podremos visualizar el código del proyecto:



Source: estudiante.js

```
1.  /**
2.   * Clase que representa a un estudiante, que es una persona y tiene una universidad asociada.
3.   * @class
4.   * @author Xiaobin Zhou
5.   */
6.  const Persona = require('./Persona');
7.
8.  class Estudiante extends Persona {
9.      /**
10.       * Constructor de la clase Estudiante.
11.       * @constructor
12.       * @param {string} nombre - El nombre del estudiante.
13.       * @param {number} edad - La edad del estudiante.
14.       * @param {string} universidad - La universidad a la que asiste el estudiante.
15.       */
16.      constructor(nombre, edad, universidad) {
17.          super(nombre, edad);
18.          this.universidad = universidad;
19.      }
20.
21.      /**
22.       * Permite al estudiante indicar que está estudiando.
23.       */
24.      estudiar() {
```


Method Details

getNombre

```
public StringⓂ getNombre()
```

Obtiene el nombre de la persona.

Returns:

El nombre de la persona.

setNombre

```
public void setNombre(StringⓂ nombre)
```

Establece el nombre de la persona.

Parameters:

nombre - El nombre a establecer.

getEdad

```
public int getEdad()
```

Obtiene la edad de la persona.

Returns:

La edad de la persona.