JavaScript

NPM es un gestor de paquetes de JavaScript que nos ayuda a instalar dependencias.

Se descarga desde https://nodejs.org/es/

- Para ver si esta instalado npm -v .
- Para crear un nuevo proyecto npm init
    - Le asignamos un nombre
    - Le asignamos versión, dejamos al principio la por defecto
    - Le asignamos la descripción
    - Le asignamos el repositorio github
    - Le asignamos autor
    - Le asignamos tipo de licencia
    - Por último “yes”
- Después de todo esto observamos que crea, dentro de la carpeta, el package.json
- Descargamos jQuery con npm install jquery
- Descargamos Bootstrap con npm install --save bootstrap
- Con el fichero package.json solo lo llevamos a otro proyecto y haciento npm init se instalan todas las dependencias especificadas
- npm install http-serve --save --dev para instalar un servidor local para el desarrollo
- npm install http-serve -g para instalar el servidor de forma global, no por proyecto
- npm install http-serve -p 4200 para indicar el puerto donde está el proyecto
- npm search angular para buscar las dependencias de angular

JSDOC

Es una herramienta de JavaScript y una biblioteca de node para documentar código de JavaScript de forma automática.

Genera a partir de comentarios la documentación.

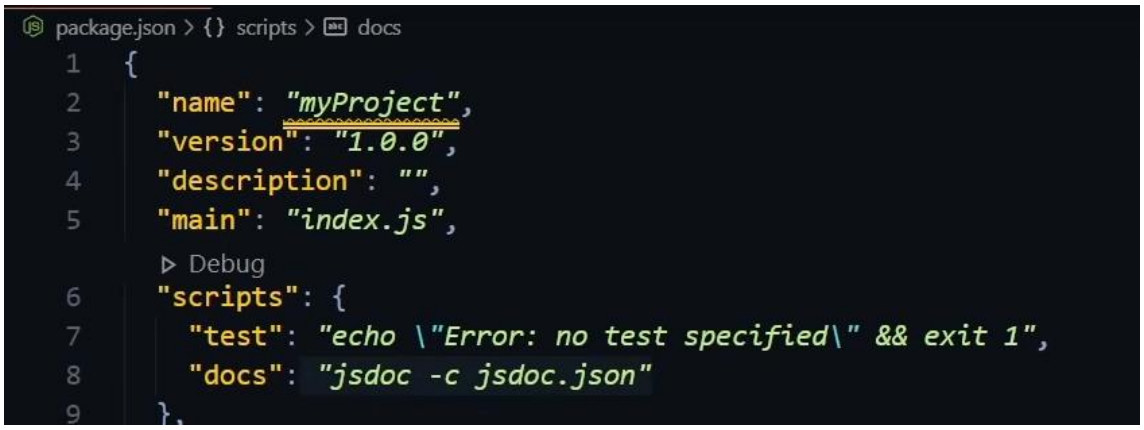
Pasos:

- crear nueva carpeta para el proyecto
- abrirlo con VSC
- la documentación se encuentra en https://jsdoc.app/
- y también en https://www.npmjs.com/package/jsdoc
- en la terminal de VSC escribimos npm init -y para crear un nuevo proyecto node
- instalamos JSDoc con el comando npm i jsdoc
- añadimos un fichero en nuestro proyecto que lo llamaremos jsdoc.json
- añadimos la configuración de jsdoc en el fichero jsdoc.json

```
{
```

```
  "plugins": [],
  "source": {
    "include": ["src"],
    "includePattern": ".js$",
    "excludePattern": "(node_modules|docs)"
  },
  "templates": {
    "cleverLinks": false,
    "monospaceLinks": false
  },
  "opts": {
    "recurse": "true",
    "destination": "./docs"
  }
}
```

- Indicamos el fichero de configuración a jsdoc para generar -> jsdoc -c jsdoc.json
- Si falla porque no tenemos instalado jsdoc globalmente si no localmente podemos lanzar el comando npx jsdoc -c jsdoc.json. Este comando creara dentro de nuestro proyecto una carpeta llamada doc que contendrá fonts, scripts, styeles y un index.html que será nuestra página de documentación
- Jsdoc también se podría ejecutar, sin ejecutar los comandos, indicando en el fichero package.json el comando y ejecutando npm run docs



- Comentario en jsdoc -> /** This is the fullname of the user */

```
/**
*This is the fullname of the user
*@type {string}
*/
cont fullName = "Juan";
```

- Abriendo en VSC “Preference: Open Workspace Settings (JSON) -> se crea una carpeta en el proyecto llamada .vscode con el fichero settings.json -> en este fichero escribimos

```
{
 “javascript”.implicitProjectConfig.checkJs”: true
}
```

- Arreglos

```
/*
* Lista de edades de usuarios
* @type {Array}
*/
const age = [19,32,53,23,12,{}. true];
/*
* Lista de edades de usuarios
* @type {Array<Number|String>}
*/
const age = [19,32,53,23,"12"];
```

- Objetos

```
/**
* Person Object
* @type {{id:number, firstName:string, lastName:string, age:number}}
*/
const persona2 = {
    id: 1,
    firstName: "Ryan",
    lastName: "Brian",
    age: 25
}
```

- Funciones

```
/**
* Add two numbers
* @param {number} n2 First Number
* @param {number} n3 Segond Number
* @returns {number} Total sum
*/
function add(n2, n3){
    return n3 + n2;
}
/**
```

```
* Add two numbers
* @param {number} n2 First Number
* @param {number} n3 Segond Number
* @returns {number} Total sum
*/
const addTwo = (n2, n3) => `The result is ${n2 + n3}`;
```

- Custom Types

```
/**
* User
* @typedef {Object} User
* @property {number} id User Id
* @property {string} name User name
* @property {number|string} [age] User age (optional)
* @property {boolean} isActive User state
*/
const myNewUser = {
    id: 1,
    name: "Laura",
    age: 30,
    isActive: true
}
/**
* @type {User}
*/
const myNewUser2 = {
    id: 2,
    name: "Tatiana",
    age: 20,
    isActive: false
}
```

- Clases

```
/* Clases */
/**
* Class to create a Programmer* @example
* const newProgrammer = new Programmer({fullName: 'Exmple Name', 'java'})
* newProgrammer.getInfo();
* @see https://google.com
* @todo Implements the rest of the methods please
```

```
*/
class Programmer{
    /**
    * @param {User} user User information
    * @param {string} Language to programming
    */
    constructor(user, language){
        this.fullName = user.fullName;
        this.language = language;
    }

    /**
    * get programmer info
    * @returns {void}
    */
    getInfo(){
        console.log("I'm ${this.fullName} and my favorite programming is ${this.language}");
    }
}
const progOne = new Programmer({fullName: "Constantin Brindusoiu"}, "javascript");
const progTwo = new Programmer({fullName: "Constantin Brindusoiu"}, "php");

progOne.getInfo();
progTwo.getInfo();
```

- @link

Min 1:02