

Material de estudio OBLIGATORIO EJE1 TypeScript

Sitio: [Instituto Superior Politécnico Córdoba](#)
Curso: Programador Web - TSDWAD - 2022
Libro: Material de estudio OBLIGATORIO EJE1 TypeScript

Imprimido por: Ezequiel Maximiliano GIAMPAOLI
Día: miércoles, 5 abril 2023, 5:41 PM

Descripción



Tabla de contenidos

1. TypeScript

2. Instalación de TypeScript

- 2.1. ¿Cómo crear y compilar un archivo TypeScript en VSCode?
- 2.2. ¿Cómo crear un servidor de pruebas en VSCode?

1. TypeScript

Introducción

Para saber TypeScript hay que conocer JavaScript.

TypeScript es un lenguaje de programación de código abierto creado por el equipo de Microsoft como una solución al desarrollo de aplicaciones de gran escala con JavaScript dado que este último carece de clases abstractas, interfaces, genéricos, etc. y demás herramientas que permiten los lenguajes de programación tipados. Son ejemplos la compatibilidad con el `intellisense`, la comprobación de tiempo de compilación, entre otras.

A typescripts se lo conoce además como un superset (superconjunto) de JavaScript ya que es un lenguaje que transpila el fuente de un lenguaje a otro pero, incluye la ventaja de que es verdaderamente orientado a objetos y ofrece además, muchas de las cosas con las que estamos habituados a trabajar los desarrolladores como por ejemplo: interfaces, genéricos, clases abstractas, modificadores, sobrecarga de funciones, decoradores, entre otras varias.

"Los superset compilan en el lenguaje estándar, por lo que el desarrollador programa en aquel lenguaje expandido, pero luego su código es "transpilado" para transformarlo en el lenguaje estándar, capaz de ser entendido en todas las plataformas"(desarrolloweb.com, <https://desarrolloweb.com/articulos/introduccion-a-typescript.html>)

Entonces, si posees algo de conocimiento de JavaScript, ¡tienes ventaja!. Podemos cambiar los archivos de JavaScript a TypeScript de a poco e ir preparando la base del conocimiento para incorporar en su totalidad este nuevo lenguaje.

Diferencias entre TypeScript y JavaScript:

JavaScript	TypeScript
JavaScript se ejecuta en el navegador. Es un lenguaje de programación interpretado.	TypeScript necesita ser "transpilado" a JavaScript, que es el lenguaje entendido por los navegadores.
JavaScript se ejecuta en el navegador, es decir en el lado del cliente únicamente.	TypeScript se ejecuta en ambos extremos. En el servidor y en el navegador.
Es débilmente tipado.	Es fuertemente tipado (tipado estático)
Basado en prototipos.	Orientado a objetos.

Tabla comparativa de los lenguajes JavaScript y TypeScript

2. Instalación de TypeScript

Instalación de TypeScript

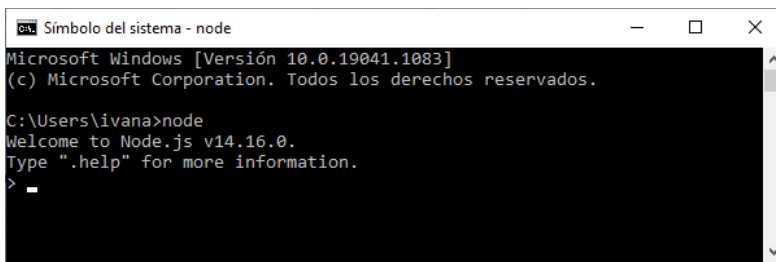
Para ejecutar código en JavaScript necesitamos instalar:

- NodeJS, dado que el compilador de Typescript está desarrollado en NodeJS.
- TSC (Command-line TypeScript Compiler), herramienta que permite compilar un archivo TypeScript a JavaScript nativo.

¿Cómo instalar NodeJS?

Para ello, seguir los siguientes pasos:

1. Descargar del sitio oficial <https://nodejs.org/es/> el instalador acorde a su sistema operativo.
2. Instalar **NodeJs**.
 1. Si tienes Windows o Mac, simplemente ejecuta el instalador y ¡listo!
 2. Si tienes Linux, la página de instalación de NodeJS ofrece los comandos para instalar en Linux. Es relativamente sencillo.
3. Evaluar que la instalación fue exitosa. Para ello, ir a la línea de comandos del sistema e introducir el comando: **node**. A continuación, el sistema *mostrará por pantalla la versión de NodeJS instalada* como sigue:



```
Símbolo del sistema - node
Microsoft Windows [Versión 10.0.19041.1083]
(c) Microsoft Corporation. Todos los derechos reservados.

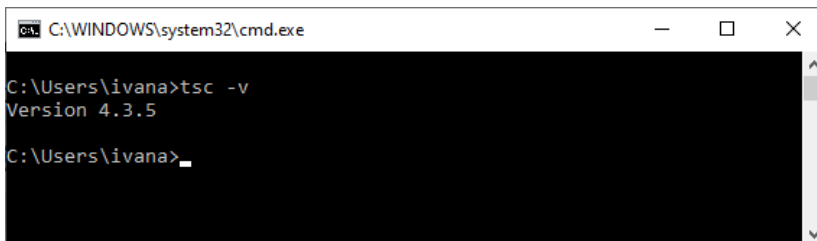
C:\Users\ivana>node
Welcome to Node.js v14.16.0.
Type ".help" for more information.
>
```

Figura 1: Línea de comandos del sistema Windows después de ejecutar el comando "node".

¿Cómo instalar TSC (Command-line TypeScript Compiler)?

La misma se realizará vía comando **npm** como sigue:

1. Abrir la línea de comandos del sistema.
2. Ejecutar el siguiente comando: **npm install -g typescript**
3. Evaluar que la instalación fue exitosa. Para ello ejecutar el comando: **tsc -v**



```
C:\WINDOWS\system32\cmd.exe
C:\Users\ivana>tsc -v
Version 4.3.5
C:\Users\ivana>
```

Figura 2: Línea de comandos del sistema luego de ejecutar el comando tsc -v

2.1. ¿Cómo crear y compilar un archivo TypeScript en VSCode?

1. Desde el explorador de archivos, crear un nuevo archivo titulado: **app.ts** dentro de una carpeta app (opcional) como sigue:

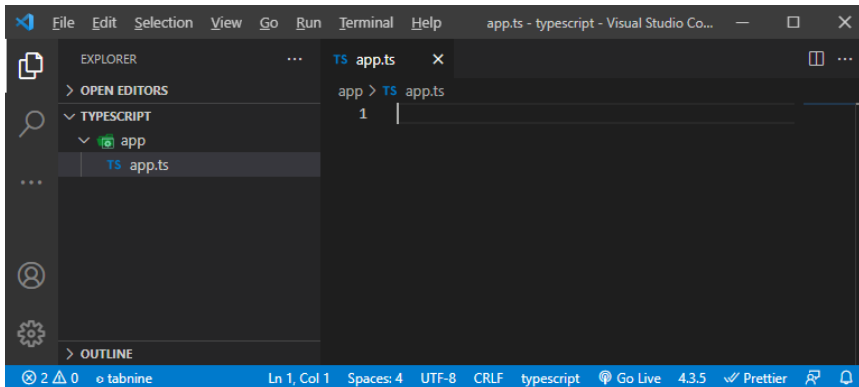


Figura 3: Creación del archivo helloworld.ts

2. Luego, escribir el código TypeScript en dicho archivo:

```
let message: string = 'Hello World';
```

```
console.log(message);
```

3. Finalmente, haciendo uso de la terminal de VSCode ejecutar el comando: **tsc app/app.ts**. Este comando compila y si no hay ningún error, crea el nuevo archivo js.

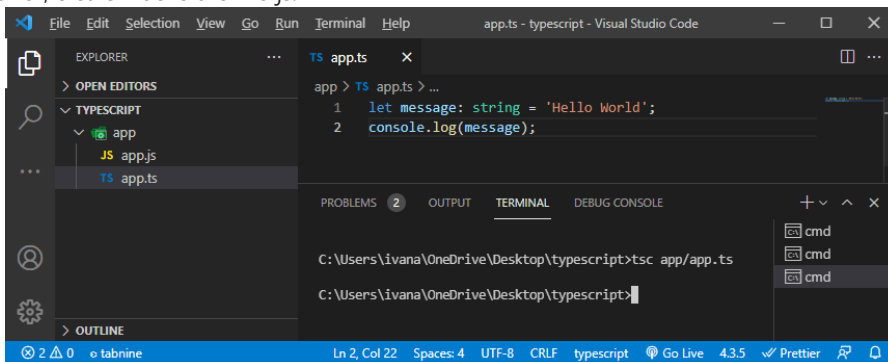


Figura 4: Compilación de TypeScript a JavaScript

Opciones del compilador

“Las opciones del compilador permiten controlar cómo se genera el código JavaScript a partir del código TypeScript de origen. Puedes establecer las opciones en el símbolo del sistema, como haría en el caso de muchas interfaces de la línea de comandos, o en un archivo JSON denominado `tsconfig.json`.

Hay disponibles numerosas opciones del compilador. Puedes ver la lista de opciones en la lista completa de opciones en la [documentación de las interfaces de la línea de comandos de tsc.](https://docs.microsoft.com/es-es/learn/modules/typescript-get-started/5-typescript-compiler) (<https://docs.microsoft.com/es-es/learn/modules/typescript-get-started/5-typescript-compiler>)

Para modificar el comportamiento predefinido del TSC con VSCode:

1. Abrir la terminal.
2. Ejecutar el comando: **tsc --init**. A continuación, se creará el archivo **tsconfig.json** como sigue:

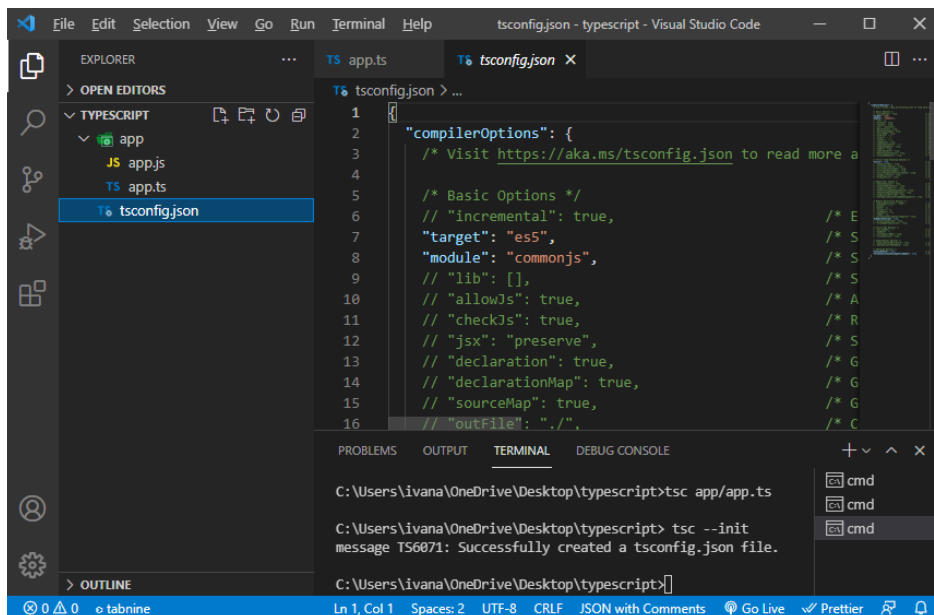


Figura 5: Archivo tsconfig.json generado después de ejecutar el comando `tsc --init`

3. Editar las configuraciones según se requiera.

Por ejemplo, podemos crear una carpeta que contenga todos los archivos .js generados por el compilador TSC (el output dir/file). Para ello, descomentar la entrada "outFile" y a continuación ejecutar como sigue:

"outFile": "../output/app.js",

y comentamos la entrada "module":

// "module": "commonjs",

y finalmente, ejecutar en la terminal de VSCode el comando **"tsc"**. A continuación, se creará la carpeta **output** conteniendo el archivo **app.js**.

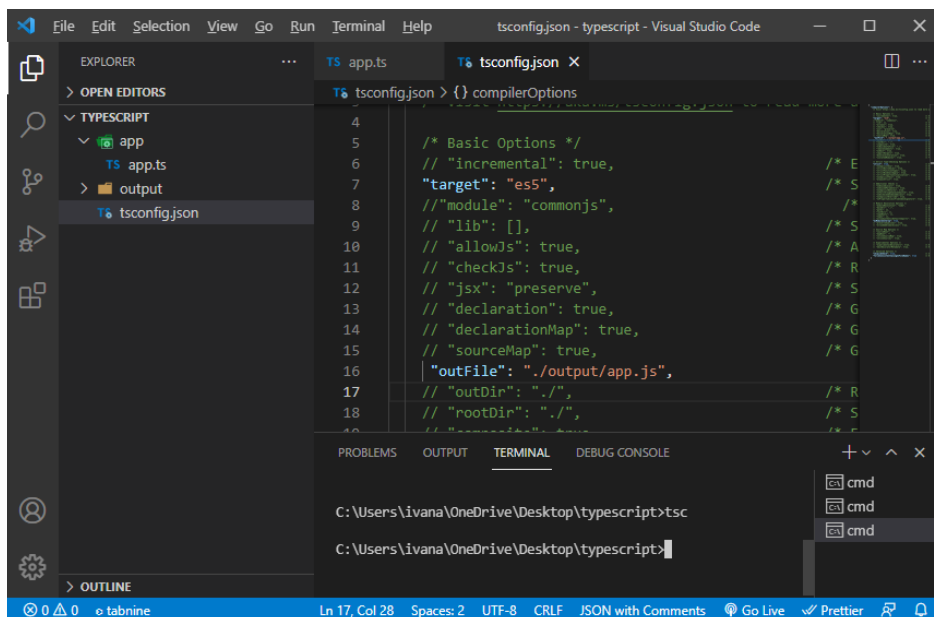


Figura 6: Manipulando la configuración del TSC para que tire los archivos generados a una carpeta output.

Bien, hasta el momento hemos creado un archivo **app.ts** y lo hemos transpilado a un archivo equivalente en JavaScript **app.js** usando el compilador TSC pero, *¿Cómo podemos ejecutarlo para ver los resultados en la consola?*

Para ello, realizar los siguientes pasos:

1. Crear un archivo **html** que incluya el script **app.js** como sigue:

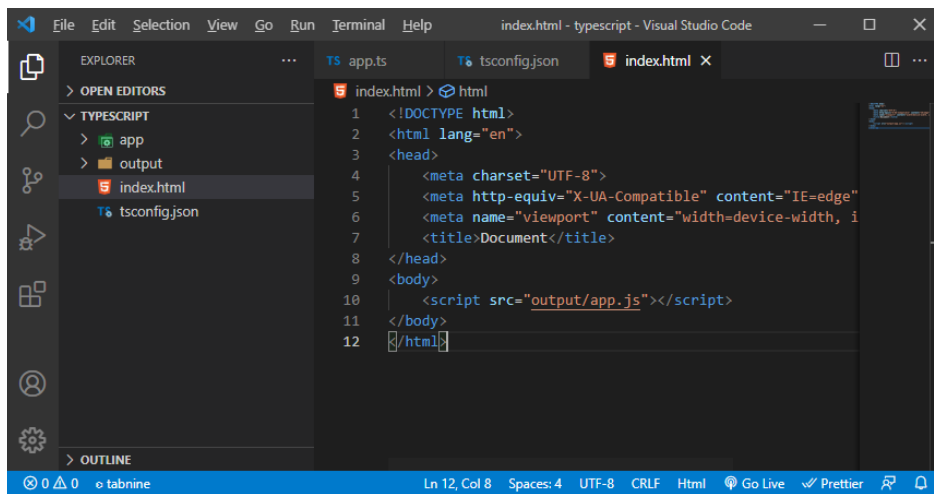


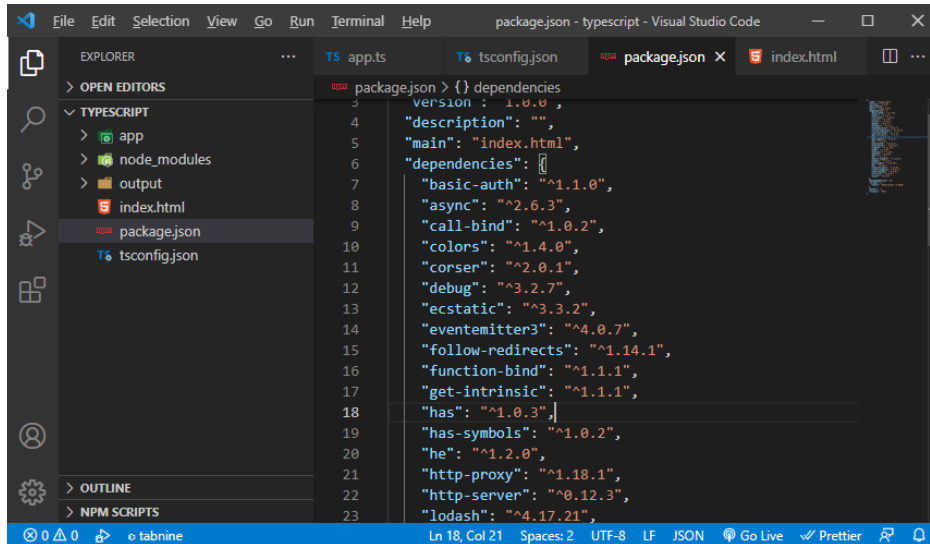
Figura 7: Archivo index.html

2. Ejecutar el archivo **index.html** o configurar un servidor de prueba para el entorno de desarrollo.

2.2. ¿Cómo crear un servidor de pruebas en VSCode?

Para ello, seguir los siguientes pasos:

1. Ejecutar el siguiente comando **"npm install --global http-server"**. A continuación, se creará la carpeta **node_modules**.
2. Ejecutar el comando **"npm init"**. A continuación, se creará un archivo **package.json**



```
package.json > {  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.html",  
  "dependencies": {  
    "basic-auth": "^1.1.0",  
    "async": "^2.6.3",  
    "call-bind": "^1.0.2",  
    "colors": "^1.4.0",  
    "corser": "^2.0.1",  
    "debug": "^3.2.7",  
    "ecstatic": "^3.3.2",  
    "eventemitter3": "^4.0.7",  
    "follow-redirects": "^1.14.1",  
    "function-bind": "^1.1.1",  
    "get-intrinsic": "^1.1.1",  
    "has": "^1.0.3",  
    "has-symbols": "^1.0.2",  
    "he": "^1.2.0",  
    "http-proxy": "^1.18.1",  
    "http-server": "^0.12.3",  
    "lodash": "^4.17.21",  
  },  
}
```

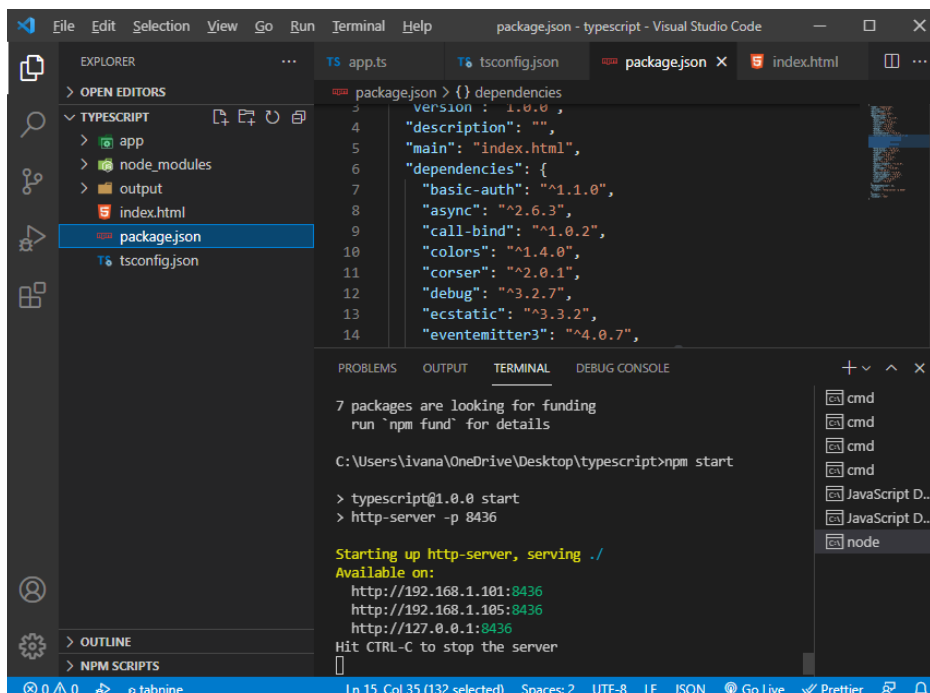
Figura 8: Archivo package.json

Nota: El archivo **package.json** es un archivo que contiene todos los metadatos acerca del proyecto. Son ejemplos: descripción, licencia, autor, dependencias, scripts, entre otros.

3. Configurar la entrada **"scripts"** como sigue:

```
"scripts": {  
  "start": "http-server -p 8456"  
}
```

4. Finalmente, ejecutar el comando **"npm start"** como sigue:



```
7 packages are looking for funding  
run 'npm fund' for details  
  
C:\Users\Ivana\OneDrive\Desktop\typescript>npm start  
  
> typescript@1.0.0 start  
> http-server -p 8436  
  
Starting up http-server, serving ./  
Available on:  
http://192.168.1.101:8436  
http://192.168.1.105:8436  
http://127.0.0.1:8436  
Hit CTRL-C to stop the server
```

Figura 9: Iniciando el servidor.

Como podemos observar en la Terminal de VSCode, accediendo a la url: <http://127.0.0.1:8436> podremos visualizar nuestro html y, si inspeccionamos el fuente el mensaje "Hola Mundo" en la consola.

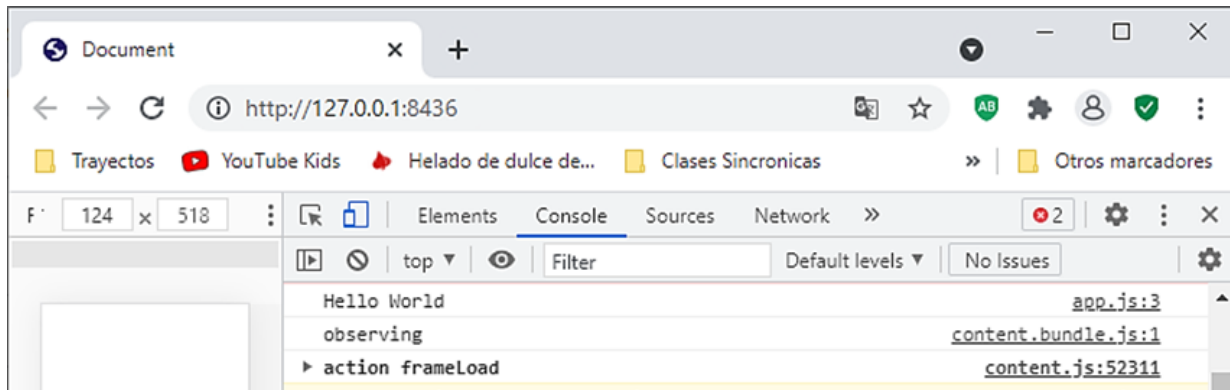


Figura 10: Inspección de código index.html