

Tutorial de un “Hola Mundo” con TypeScript

El siguiente tutorial trata sobre cómo lograr escribir un hola mundo utilizando Typescript, abarca desde la instalación de las herramientas necesarias hasta el código que se requerido.

Existen dos formas de poder hacer uso de TypeScript:

- Via npm (Node.js package manager).
- Instalando los plugins de TypeScript en Visual Studio

Para efectos de este tutorial se estará siguiendo el proceso de instalación por medio de npm, a continuación, se detallan los pasos:

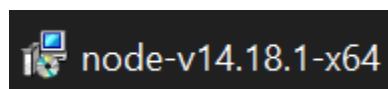
- 1- Para poder hacer uso de npm hay que instalar Node.js, el cual se puede descargar desde el propio sitio de node.js:

<https://nodejs.org/en/download/>

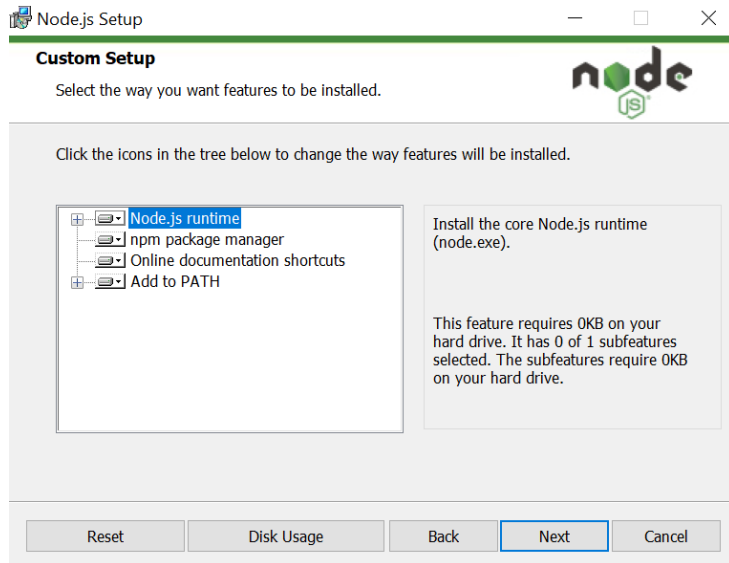
	LTS Recommended For Most Users	Current Latest Features
Windows Installer (.msi)	node-v14.18.1-x64.msi	
macOS Installer (.pkg)	node-v14.18.1.pkg	
Source Code		node-v14.18.1.tar.gz

	32-bit	64-bit
Windows Installer (.msi)		
Windows Binary (.zip)		
macOS Installer (.pkg)	64-bit	
macOS Binary (.tar.gz)	64-bit	

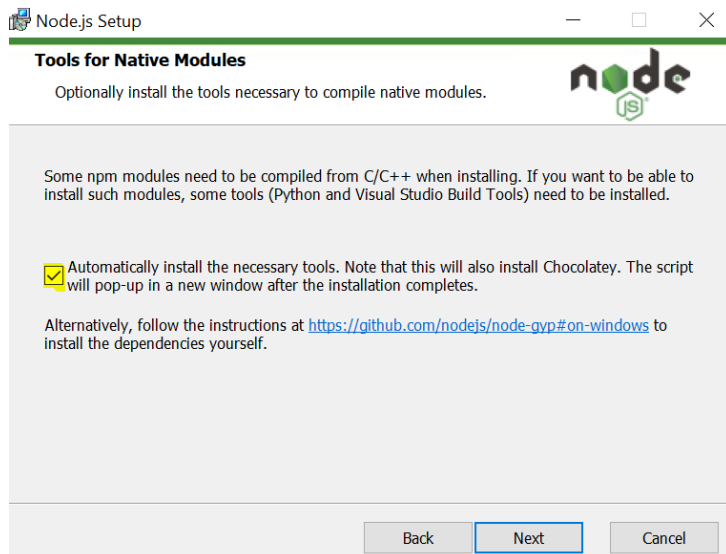
- Para usuarios de Windows seleccione la opción LTS (long-term support) para Windows, en caso de ser usuario de mac, seleccione la opción LTS para mac.
- El archivo descargado es el siguiente:



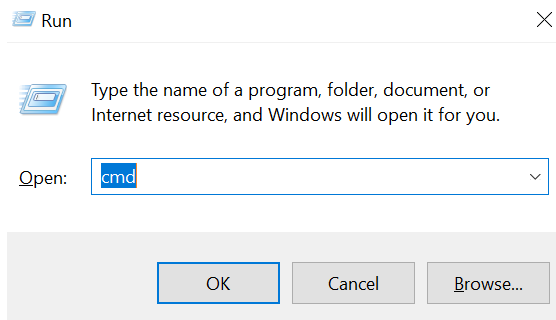
- Al ejecutar el archivo descargado aparecerá setup al que le damos next:



- A continuación, en la siguiente pantalla que aparece le damos check al checkbox para que automáticamente instale todas las herramientas necesarias



- Finalmente se procede con la instalación en la siguiente pantalla y lo que resta es solo esperar a que la instalación finalice, la cual tardara unos minutos mientras
- Cuando la instalación haya finalizado podemos hacer una verificación de si la instalación se concretó con éxito haciendo uso del CMD de Windows, presionando el botón de Windows + R:



- Ya en el CMD hacemos hacer la verificación de la versión de Node.js instalada haciendo uso del comando “node -v”:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18363.1801]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\obarb>node -v
v14.18.1

C:\Users\obarb>
```

- Para comprobar si se instalo correctamente el Node Package Manager, verificamos la versión instalada con el comando “npm -v”

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18363.1801]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\obarb>npm -v
6.14.15

C:\Users\obarb>
```

- Con estos últimos pasos confirmamos que Node.js fue instalado exitosamente y ya podremos hacer uso del instalador de paquetes NPM para poder hacer uso de TypeScript.

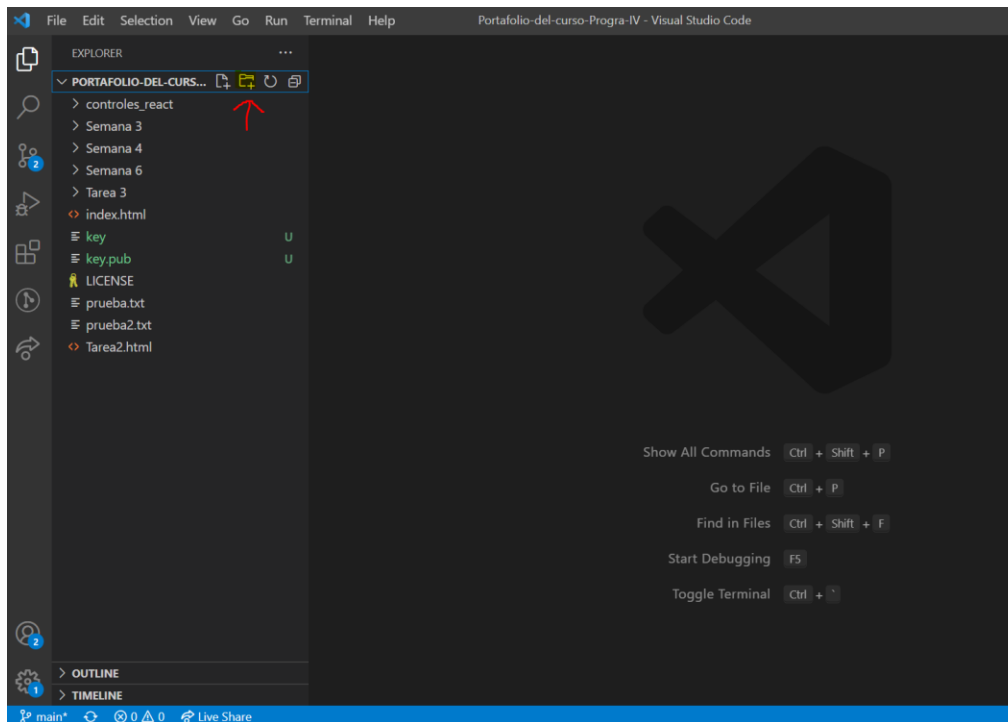
- 2- Teniendo debidamente instalado Node.js podemos instalar TypeScript utilizando NPM con el comando “npm install -g typescript” como se muestra a continuación:

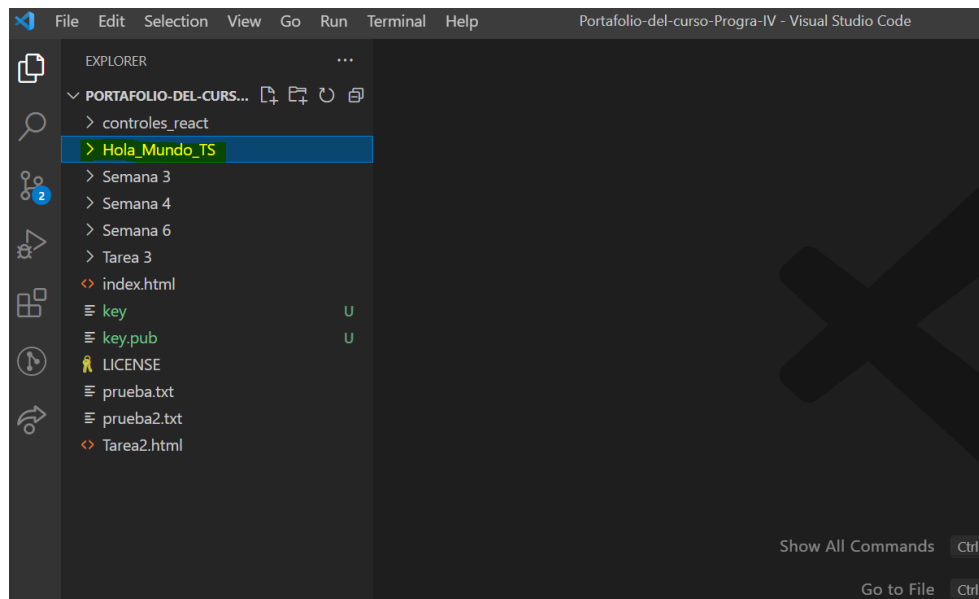
```
C:\Users\obarb>npm install -g typescript
C:\Users\obarb\AppData\Roaming\npm\tsserver -> C:\Users\obarb\AppData\Roaming\npm\node_modules\typescript\bin\tsserver
C:\Users\obarb\AppData\Roaming\npm\tsc -> C:\Users\obarb\AppData\Roaming\npm\node_modules\typescript\bin\tsc
+ typescript@4.4.4
added 1 package from 1 contributor in 5.078s
```

- 3- Diríjase a la ubicación de su portafolio y asegúrese que este actualizado haciendo un “git pull”, esto para evitar que haya conflictos posteriores al realizar un commit de lo que se realice a continuación:

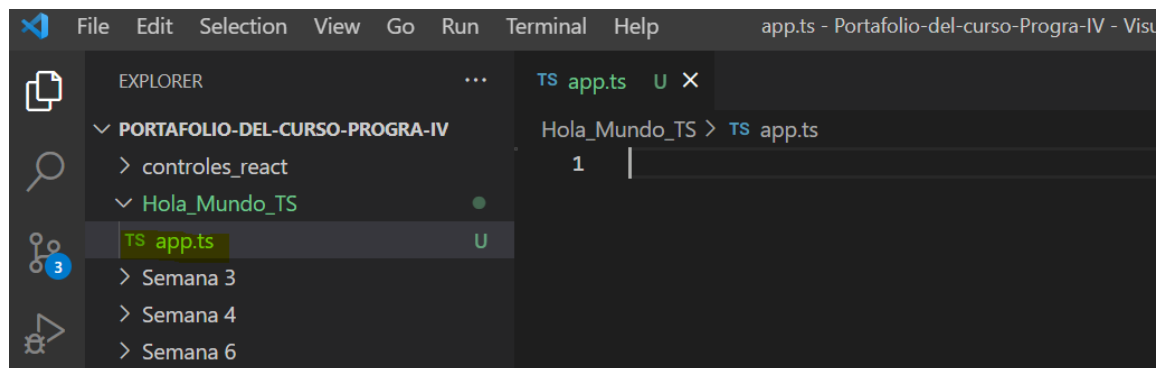
```
OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ git pull
Already up to date.
```

- 4- Ahora procedemos a realizar nuestro Hola Mundo, para esto utilizaremos Visual Studio Code, y dentro de nuestro portafolio creamos una nueva carpeta con el nombre “Hola_Mundo_TS”.

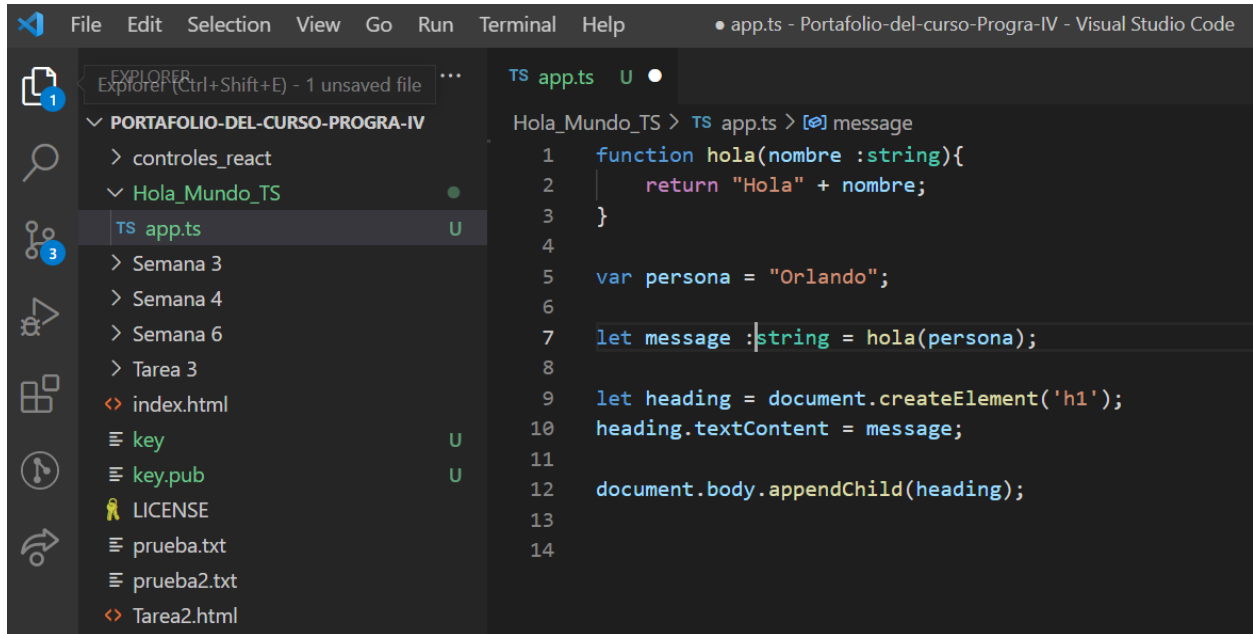




5- Dentro de la Carpeta creamos un archivo Typescript llamado “app.ts”:



6- Dentro de este archivo ingresamos las siguientes líneas y guardamos los cambios:



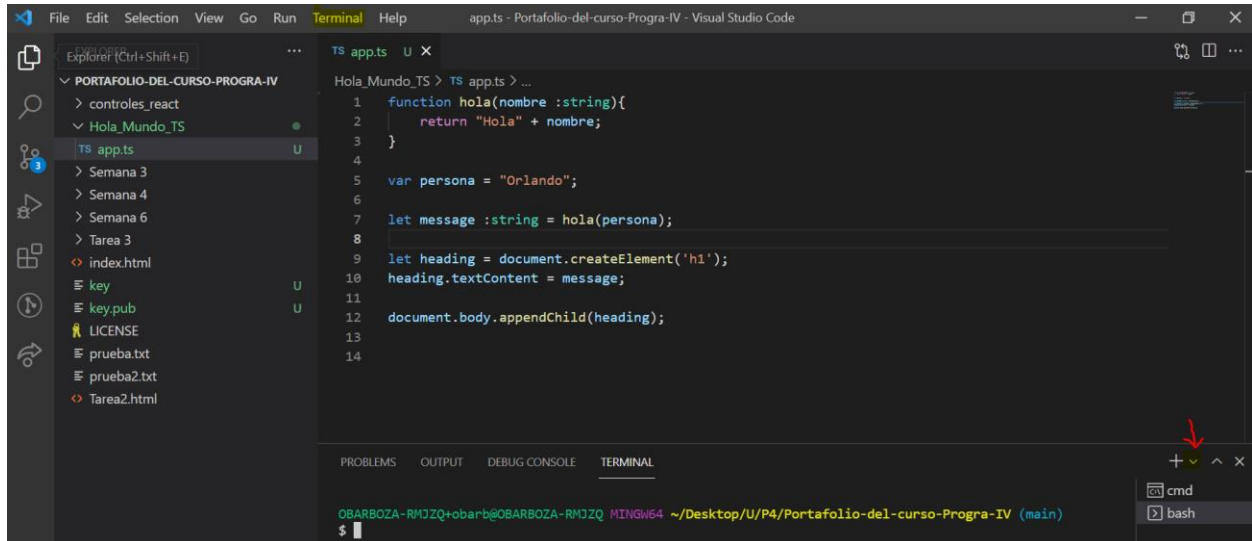
```
File Edit Selection View Go Run Terminal Help • app.ts - Portafolio-del-curso-Progra-IV - Visual Studio Code

EXPLORER Explorer (Ctrl+Shift+E) - 1 unsaved file TS app.ts U
PORTAFOLIO-DEL-CURSO-PROGRA-IV
  > controles_react
  > Hola_Mundo_TS
    TS app.ts U
    > Semana 3
    > Semana 4
    > Semana 6
    > Tarea 3
    > index.html
    > key U
    > key.pub U
    > LICENSE
    > prueba.txt
    > prueba2.txt
    > Tarea2.html

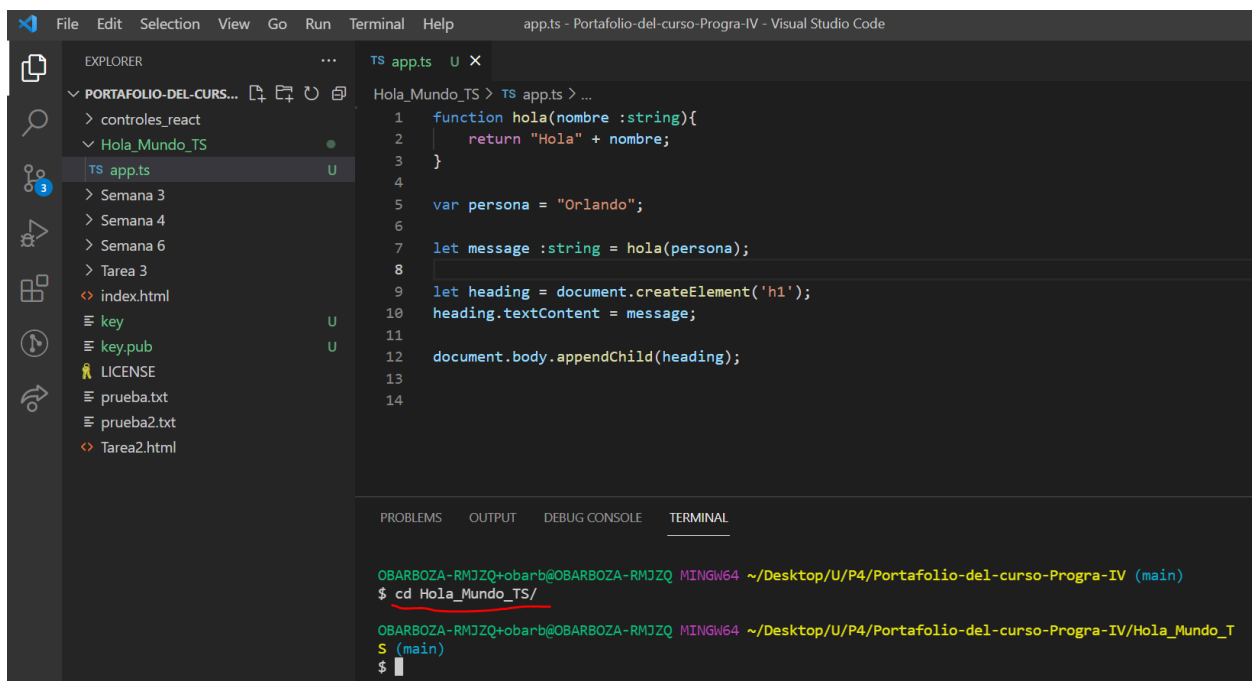
Hola_Mundo_TS > TS app.ts > [⌘] message
1 function hola(nombre :string){
2   return "Hola" + nombre;
3 }
4
5 var persona = "Orlando";
6
7 let message :string = hola(persona);
8
9 let heading = document.createElement('h1');
10 heading.textContent = message;
11
12 document.body.appendChild(heading);
13
14
```

- En la primera parte creamos una función con el nombre “hola”, donde por parámetros le ingresaremos un string, aquí se puede observar como por medio de Typescript especificamos el tipo de dato a ingresar.
- Después creamos una variable llamada “persona” donde guardaremos un nombre.
- Declaramos una variable de nombre “message”, le especificamos que es de tipo string y le asignamos el valor de la función creada previamente.
- Agregamos un nuevo encabezado de tipo h1 y le agregamos el contenido de la variable “message”.
- Finalmente agregamos el encabezado al documento.

- 7- Abrimos una terminal en Visual Studio Code por medio del menú superior llamado “terminal”, para poder compilar el código y seleccionamos una terminal bash según se muestra a continuación siguiendo la flecha roja:



- 8- Navegamos a la carpeta “Hola_Mundo_TS” que contiene el archivo que deseamos compilar:



- 9- Compilamos el archivo “app.ts” usando el comando “tsc app.ts” y podremos observar cómo se crea un archivo del mismo nombre, pero con la extensión .js:

The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left, the Editor in the center, and the Terminal at the bottom. The Explorer sidebar shows a project structure with a folder named 'HOLA_MUNDO_TS' containing a file 'app.ts'. The Editor shows the content of 'app.ts', which is a TypeScript file defining a function 'hola' and using it to create and append an element to the document body. The Terminal shows the command 'tsc app.ts' being executed, which compiles the TypeScript file into a JavaScript file named 'app.js'.

```
File Edit Selection View Go Run Terminal Help app.ts - Portafolio-del-curso-Progra-IV - Visual Studio Code
```

EXPLORER

- PORTAFOLIO-DEL-CURSO-PROGRA-IV
 - controles_react
 - HOLA_MUNDO_TS
 - app.ts
 - Semana 3
 - Semana 4
 - Semana 6
 - Tarea 3
 - index.html
 - key
 - key.pub
 - LICENSE
 - prueba.txt
 - prueba2.txt
 - Tarea2.html

TS app.ts

```
1 function hola(nombre :string){
2   return "Hola" + nombre;
3 }
4
5 var persona = "Orlando";
6
7 let message :string = hola(persona);
8
9 let heading = document.createElement('h1');
10 heading.textContent = message;
11
12 document.body.appendChild(heading);
13
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/U/P4/Portafolio-del-curso-Progra-IV (main)
$ cd HOLA_MUNDO_TS/

OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/U/P4/Portafolio-del-curso-Progra-IV/HOLA_MUNDO_TS (main)
$ tsc app.ts
```

- Si revisamos este nuevo archivo llamado “app.js” veremos que tiene el mismo código que el otro, con la diferencia que no se aprecian los tipos de datos como se especificaron en el archivo escrito en TypeScript.

The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left, the Editor in the center, and the Terminal at the bottom. The Explorer sidebar shows the project structure with a folder named 'HOLA_MUNDO_TS' containing a file 'app.js'. The Editor shows the content of 'app.js', which is the compiled JavaScript version of the TypeScript file, where type annotations have been removed. The Terminal shows the command 'cat app.js' being executed, displaying the content of the compiled file.

```
File Edit Selection View Go Run Terminal Help app.js - Portafolio-del-curso-Progra-IV - Visual Studio Code
```

EXPLORER

- PORTAFOLIO-DEL-CURS...
 - controles_react
 - HOLA_MUNDO_TS
 - app.js
 - Semana 3
 - Semana 4
 - Semana 6
 - Tarea 3

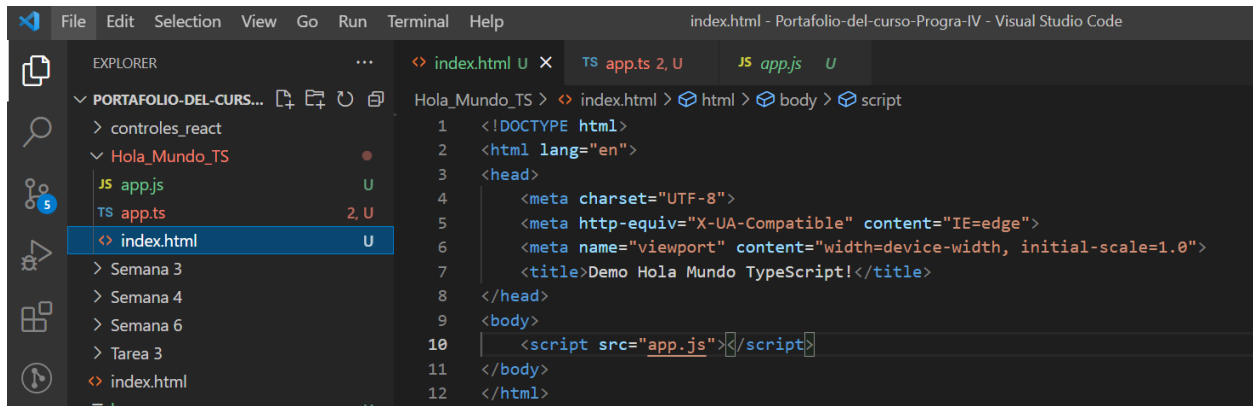
JS app.js

```
1 function hola(nombre) {
2   return "Hola" + nombre;
3 }
4
5 var persona = "Orlando";
6 var message = hola(persona);
7 heading.textContent = message;
8 document.body.appendChild(heading);
9
```

TERMINAL

```
OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/U/P4/Portafolio-del-curso-Progra-IV/HOLA_MUNDO_TS (main)
$ cat app.js
```

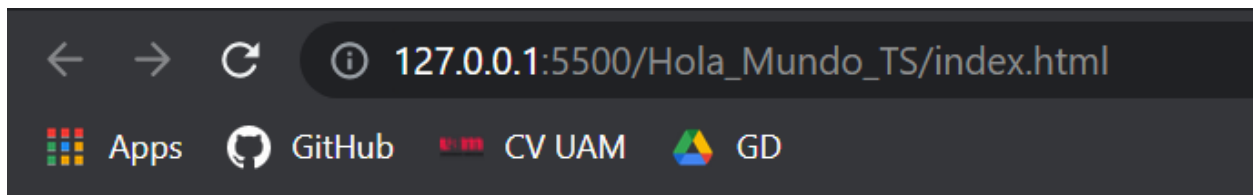
- 10- Procedemos a crear un nuevo archivo html dentro de la carpeta llamado “index.html” e ingresamos el siguiente código donde en el cuerpo incluimos el archivo que acabamos de compilar y guardamos los cambios:



The screenshot shows the Visual Studio Code interface. In the Explorer on the left, the file 'index.html' is selected under the 'Hola_Mundo_TS' folder. The editor view on the right displays the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Demo Hola Mundo TypeScript!</title>
8 </head>
9 <body>
10  <script src="app.js"></script>
11 </body>
12 </html>
```

- 11- Finalmente le damos click derecho al archivo y abrimos con live server y el resultado será el siguiente:



Hola Orlando

Para salvar el archivo en el repositorio remoto, siga los siguientes pasos:

1- Agregamos el file a staggig "git add Hola_Mundo_TS"

```
OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ git add Hola_Mundo_TS/

OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   Hola_Mundo_TS/app.js
    new file:   Hola_Mundo_TS/app.ts
    new file:   Hola_Mundo_TS/index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    key
    key.pub

OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ |
```

2- Salvamos archivo dentro del repositorio local (commit) "git commit -m "Hola mundo con typescript".

```
OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ git commit -m "Hola mundo con typescript"
[main 2ad77dc] Hola mundo con typescript
3 files changed, 31 insertions(+)
create mode 100644 Hola_Mundo_TS/app.js
create mode 100644 Hola_Mundo_TS/app.ts
create mode 100644 Hola_Mundo_TS/index.html

OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    key
    key.pub

nothing added to commit but untracked files present (use "git add" to track)

OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ |
```

3- Lo subimos al repositorio remoto "git push origin main"

```
OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
$ git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 840 bytes | 120.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To github.com:obarboza92/Portafolio-del-curso-Progra-IV.git
   b4fc03a..2ad77dc  main -> main

OBARBOZA-RMJZQ+obarb@OBARBOZA-RMJZQ MINGW64 ~/Desktop/u/P4/Portafolio-del-curso-Progra-IV (main)
```