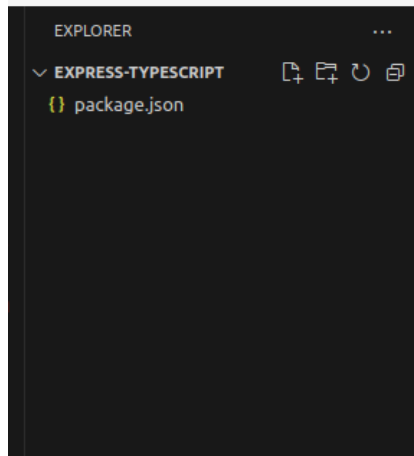




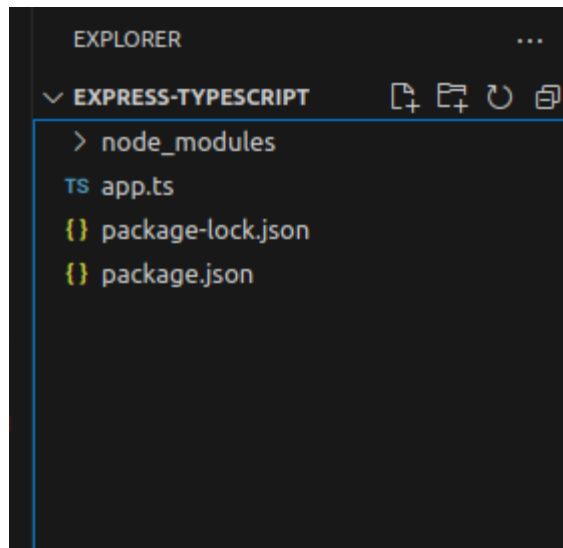
CÓMO USAR EXPRESS CON TYPESCRIPT

Requisitos: Tener instalado nodejs <https://nodejs.org/>

1. Creamos la carpeta express-typescript donde trabajaremos nuestro proyecto, empezamos como lo hacemos habitualmente para trabajarlo con Javascript, usamos *npm init* para iniciarlo.



2. Creamos el archivo app.ts e Instalamos Express y Dotenv(para manejar variables de entorno) con *npm i express dotenv*



3. Instalar typescript como dependencia de desarrollo del proyecto: *npm i -D typescript*
4. Instalar express typescript como dependencia de desarrollo del proyecto: *npm install -D @types/express*
5. Instalar dotenv como dependencia de desarrollo del proyecto: *npm install -D @types/dotenv*



6. Creamos nuestro proyecto typescript usando `npx tsc --init` En el archivo `tsconfig.json` generado, habilitamos el directorio de salida `./dist` En esta carpeta tendremos nuestro proyecto Express transpilado a `.js`

Una captura de pantalla del editor de código Visual Studio Code. El panel de la izquierda muestra el explorador de archivos con la carpeta "EXPRESS-TYPESCRIPT" seleccionada, mostrando archivos como "node_modules", "app.ts", "package-lock.json", "package.json" y "tsconfig.json". El panel principal muestra el contenido del archivo "tsconfig.json" con la siguiente configuración:

```

{
  "compilerOptions": {
    // "customConditions": [],
    // "resolveJsonModule": true,
    // "allowArbitraryExtensions": true,
    // "noResolve": true,

    /* JavaScript Support */
    // "allowJs": true,
    // "checkJs": true,
    // "maxNodeModuleJsDepth": 1,

    /* Emit */
    // "declaration": true,
    // "declarationMap": true,
    // "emitDeclarationOnly": true,
    // "sourceMap": true,
    // "inlineSourceMap": true,
    // "outFile": "./",
    "outDir": "./dist",
    // "removeComments": true,
    // "noEmit": true,
    // "importHelpers": true,
    // "importsNotUsedAsValues": "remove",
    // "downlevelIteration": true,
    // "sourceRoot": "",
    // "mapRoot": "",
    // "inlineSources": true.
  }
}

```

7. Agregamos el siguiente código de nuestro server al archivo `app.ts`:



```
import express, { Request, Response } from "express";
import dotenv from "dotenv";

dotenv.config();

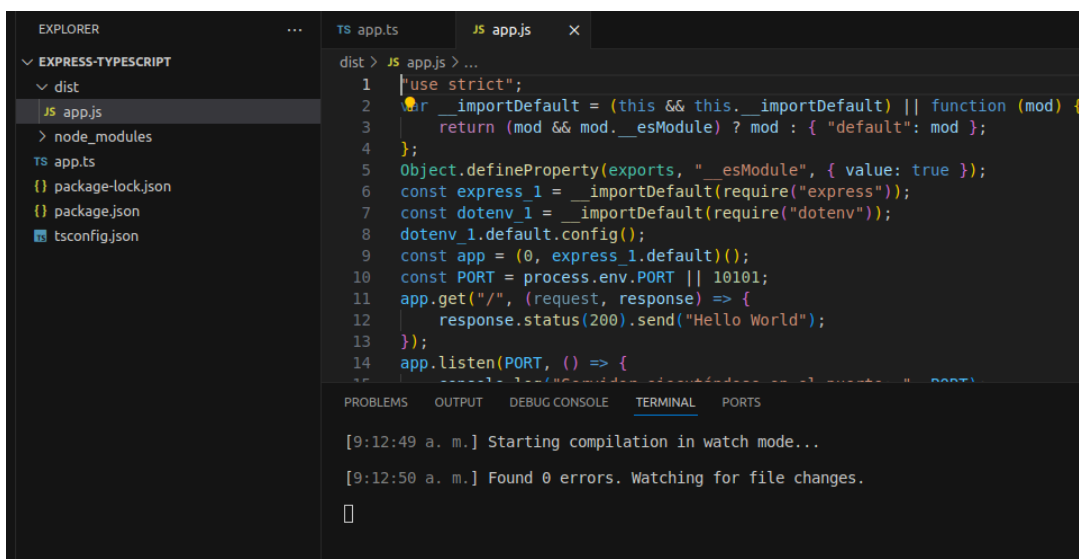
const app = express();

const PORT = process.env.PORT || 10101;

app.get("/", (request: Request, response: Response) => {
  response.status(200).send("Hello World");
});

app.listen(PORT, () => {
  console.log("Servidor ejecutándose en el puerto: ", PORT);
}).on("error", (error) => {
  throw new Error(error.message);
});
```

8. ejecutamos `npx tsc -w` en la consola/terminal del proyecto para activar el modo observador de Typescript. Vemos como se crea la carpeta `dist`, en ella encontraremos nuestro código transpilado a `.js`





9. Creamos otra terminal/consola en nuestro proyecto y ejecutamos nuestro server con el comando `node ./dist/app`

La imagen muestra una interfaz de desarrollo (VS Code) con dos paneles principales. El panel superior es el editor de código, que muestra un archivo `app.ts` con el siguiente contenido:

```
1 import express, { Request, Response } from "express";
2 import dotenv from "dotenv";
3
4 dotenv.config();
5 const app = express();
6
7 const PORT = process.env.PORT || 10101;
8
9 app.get("/", (request: Request, response: Response) => {
10   response.status(200).send("Hello World");
11 });
12
13 app.listen(PORT, () => {
14   console.log("Servidor ejecutándose en el puerto: ", PORT);
15 });
```

El panel inferior es la terminal, que muestra el comando `node ./dist/app` ejecutado y el mensaje de salida: `Servidor ejecutándose en el puerto: 10101`.

10. Vamos a nuestro navegador a <http://localhost:10101/> y verificamos que nuestro server trabaja.

