TYPESCRIPT-INTRODUCTION
// Book : Beginning TypeScript *By Nathan Sebhastian 2023*

What is TypeScript?

TypeScript is an open-source programming language that adds a typing system to JavaScript.

Basically, TypeScript is an extension of JavaScript. It has the same syntax as JavaScript.
It has all of JavaScript features, and every JavaScript .js file is a valid TypeScript file.

TypeScript can't be executed on JavaScript environment such as the browser or Node.js.
When you want to deploy the application, you need to transform TypeScript into JavaScript using the TypeScript compiler program.

When you use TypeScript, you essentially turn JavaScript into a static type language.
In a statically typed language, the type of every single variable used in your code is known before you even run the code.
You can't change the type of your variables after it has been declared, except by explicitly converting them.

When to Use TypeScript?

TypeScript is usually used on medium to large projects, where there's a team of developers maintaining and developing features on a daily basis.

But in all honesty, you can use TypeScript whenever you want because TypeScript has an incremental adoption strategy baked into the language itself. This means you can convert a JavaScript project into a TypeScript project incrementally, step by step as you need it.

TYPESCRIPT Project ---
// a folder my-ts-project

First, you need to install the *TypeScript compiler* using npm. On your terminal, run the command below:

```
npm install -g typescript
```

The -g option is short for global installation. This is used so that we can call the TypeScript compiler from the command line directly. Once the installation is finished, you can verify that the package is installed correctly by checking its version:

```
tsc -v
```

Next, create a folder my-ts-project that will be used to store all files Name that file as index.ts because .ts is the TypeScript file format.

```
// index.ts
let myNumber: number = 10;
console.log(myNumber);
```

Transpiling TypeScript into JavaScript Files ---

Open the terminal on your TypeScript folder, then run the tsc command followed by the .ts file name as follows:

```
tsc index.ts
```

TypeScript still generates an *index.js* file. Now this is the usual JavaScript code that can be executed by JavaScript environment.

TypeScript is only used in the coding phase, and it always needs to be converted to JavaScript using the compiler before running.

TypeScript Configuration File tsconfig.json

In the generated .js file above, you can see that TypeScript uses var instead of let for the variable because it tries to keep the code compatible with older environments.

You can actually tell TypeScript to use let and const for the generated code by creating a *TypeScript configuration* in the form of a JSON file.

Back to the terminal, run the command below:

```
tsc --init
```

The --init option will initialize and create a new tsconfig.json file in your project.

```
// tsconfig.json
Created a new tsconfig.json with:
target: es2016
module: commonjs
strict: true
esModuleInterop: true
skipLibCheck: true
forceConsistentCasingInFileNames: true
```

The options above are the defaults enabled by TypeScript.

The tsc --init command generates all possible *options you can define in the JSON file*

target option is the JavaScript version for the generated code.

module option which sets the JavaScript module system.
Setting to commonjs will use the module.exports and require() syntax,
while setting to any ES version will use the export and import syntax.
If you use Node.js to run JavaScript code, it's better to leave it as commonjs. Node.js supports ES module syntax, but you have to set the type: module option in the package.json file.

rootDir option, which sets *the root directory of your source code*.
This is a useful option to separate the TypeScript and JavaScript files, so let's uncomment this option and set it to ./src as follows:

```
// tsconfig.json
"rootDir": "./src"
```

Next, create the src/ folder in your project, and move the index.ts file there.

outDir option specifies where TypeScript should generate *the output of the transpilation*.
Set it to ./dist as follows:

```
// tsconfig.json
"outDir": "./dist"
```

removeComments option so that all comments in the TypeScript files won't be preserved in JavaScript files.

noEmitOnError option, which stops the transpilation when there's an error in the TypeScript files.

With these options, now you can *run TypeScript compiler* without specifying the file:

```
tsc
```

The tsc command will look into *tsconfig.json file* and use your rootDir option to find the .ts files.

Now TypeScript will create the dist/ folder and place the generated index.js file there.
Good work!