Helpdesk Session 1

TypeScript

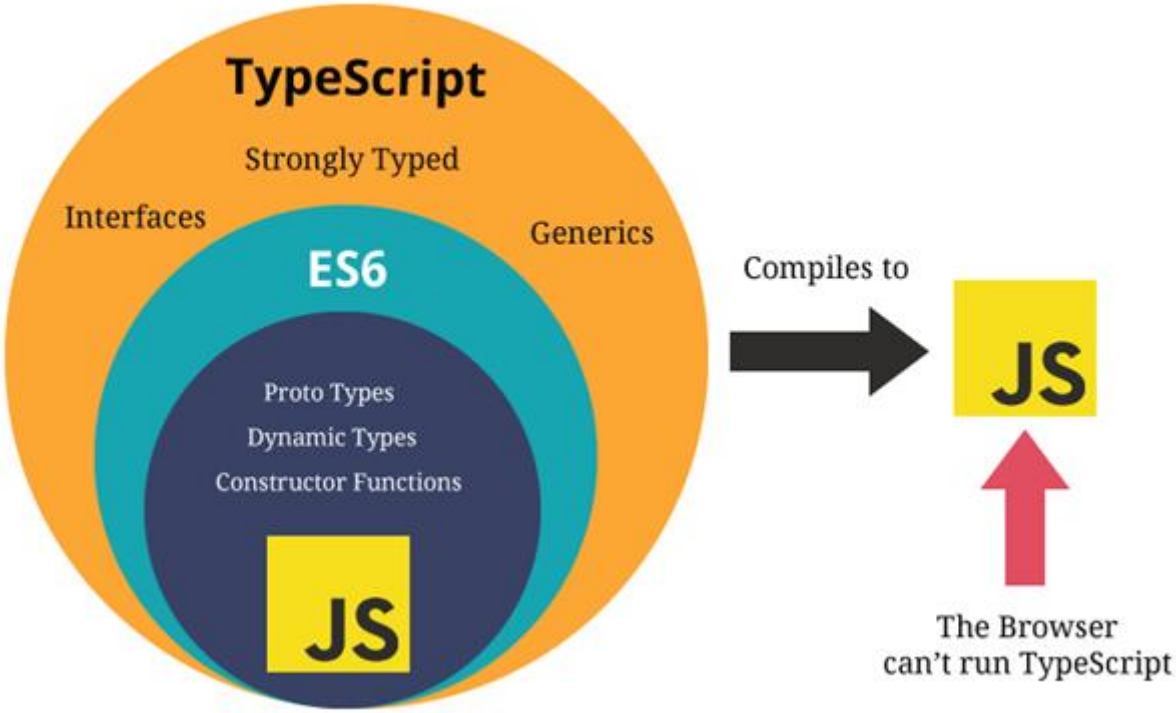
Agenda

- TypeScript Introduction
- Why should we use TypeScript?
- Development Environment Setup for TypeScript
  - Install Node.js
  - Install the TypeScript Compiler
  - Install and setup VSCode
- Overview of tsconfig.json
- Transpile TS into JavaScript
- Variables in TS
- TypeScript Simple Types
- Additional Types

TypeScript Introduction

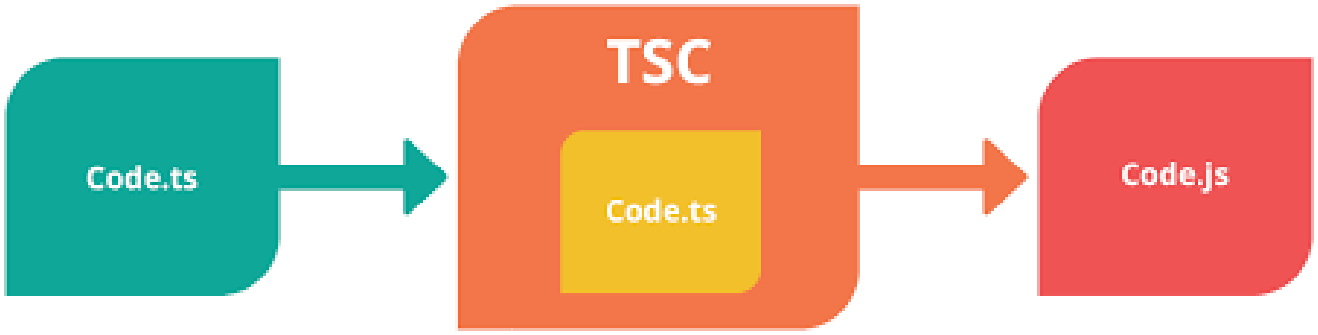
- TypeScript is a syntactic superset of JavaScript which adds static typing.

TypeScript being a "Syntactic Superset" means that it shares the same base syntax as JavaScript, but adds something to it.



TypeScript Introduction

- TypeScript doesn’t run in the browser. We need to compile it to JS to run.
- TypeScript is a typed superset of JavaScript and pure Object oriented.



TypeScript Introduction

Features of TypeScript:

- Due to the static typing, code written in TypeScript is more predictable, and is generally easier to debug.
- Makes it easier to organize the code for very large and complicated apps.
- You can use TypeScript for other JS libraries, Because TS compiles plain JS.
- TS is platform Independent.

TypeScript

Why should we use TypeScript?

- JavaScript is a loosely typed language. It can be difficult to understand what types of data are being passed around in JavaScript.
- In JavaScript, function parameters and variables don't have any information! So developers need to look at documentation, or guess based on the implementation.
- TypeScript allows specifying the types of data being passed around within the code, and has the ability to report errors when the types don't match.
- For example, TypeScript will report an error when passing a string into a function that expects a number. JavaScript will not.

Development Environment Setup for TypeScript

Install Node.js (Windows)

- Download from the link https://nodejs.org/en/download/
- Install the exe file
- Type the following command in command prompt for verification
  node -v
  npm -v

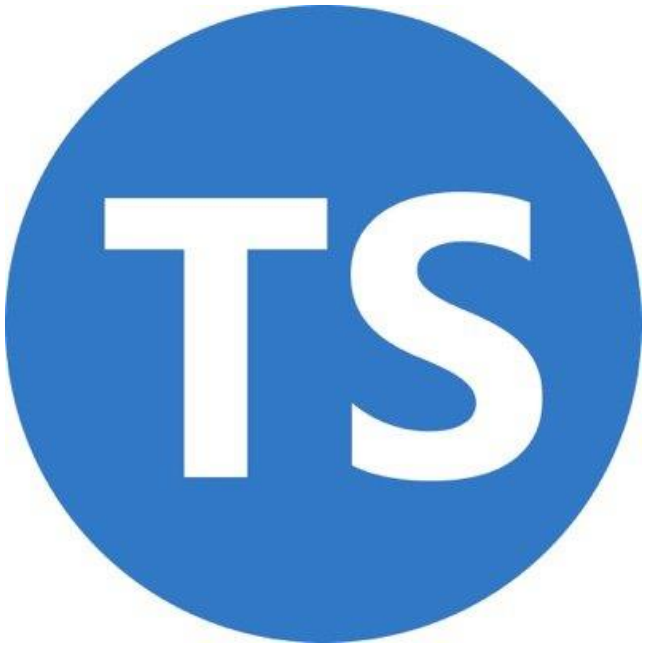
Install Node.js (Linux)

- Refresh you local package index by the command:
  sudo apt update
- Install Node.js through the command
  sudo apt install nodejs
- Type the following command in terminal for verification
  node -v
  npm -v

Development Environment Setup for TypeScript

Install the TypeScript Compiler (Windows)

- Type the command on command prompt (CMD)
  npm i -g typescript
- Type the following command in CMD for verification
  tsc -v



Install Typescript Compiler (Linux)

- Refresh you local package index by the command:
  sudo apt update
- Install Node.js through the command
  npm install -g typescript
- Type the following command in terminal for verification
  node -v

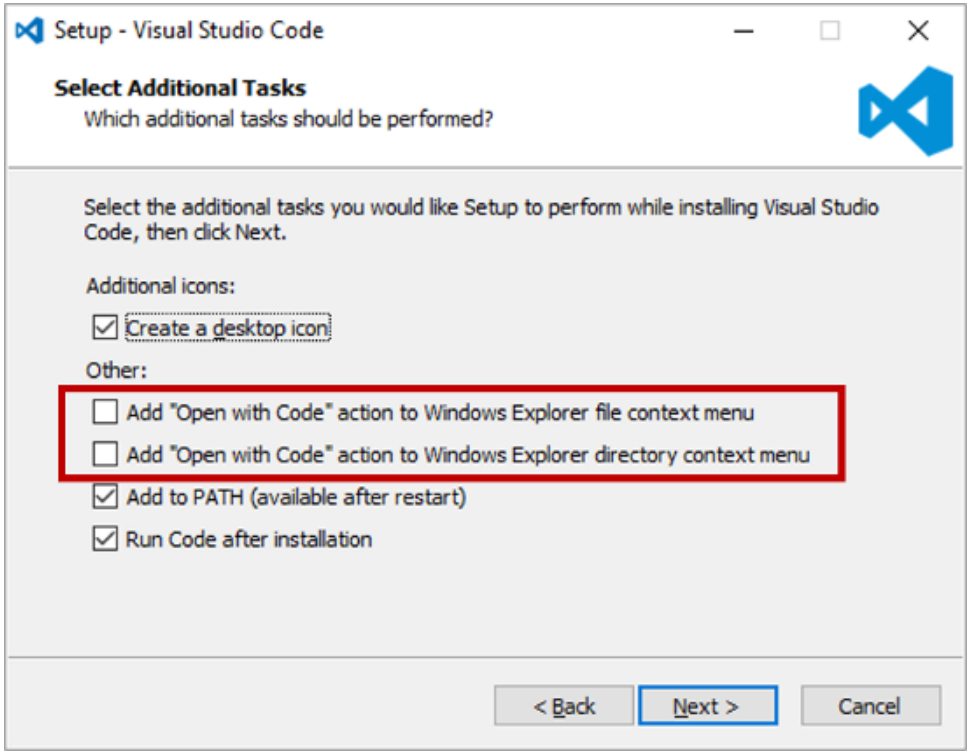
Development Environment Setup for TypeScript

Install the VS Code (Windows)

- Download the Visual Studio Code installer for Windows from https://go.microsoft.com/fwlink/?LinkID=534107
- Once it is downloaded, run the installer

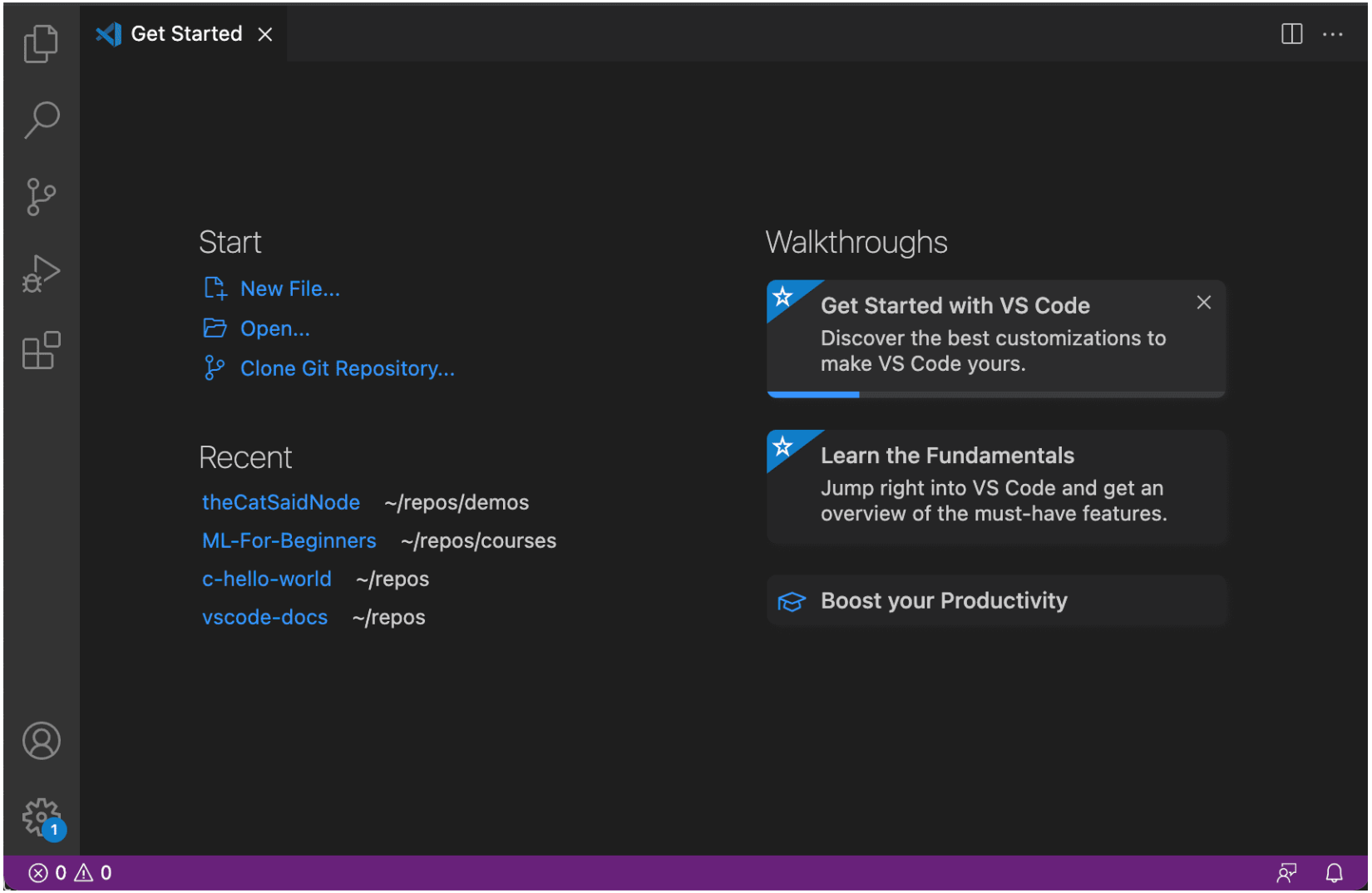
Visual Studio Code

- When you see the dialog box below, be sure to check the two checkboxes highlighted.



Development Environment Setup for TypeScript

Install the VS Code (Windows)



Overview of “tsconfig.json”

- TypeScript create **tsconfig.json** with the recommended settings when we initiate it in a folder with command:

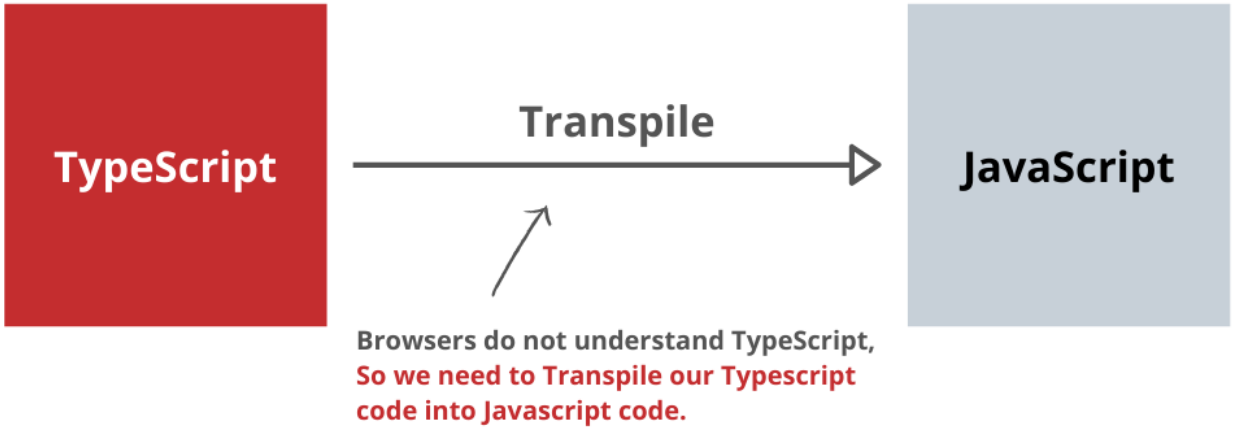
  tsc - -init

- you have been relying on the TypeScript compiler's default behavior to compile your TypeScript source code.
- You can modify the TypeScript compiler options by adding a tsconfig.json file that defines the TypeScript project settings such as the compiler options and the files that should be included.

Transpile TS into JavaScript

- VS Code integrates with **tsc** through our integrated task runner.
- Transpiling a simple TypeScript Hello World program.

Step 1: Create a simple TS file
Step 2: Run the TypeScript build



TypeScript

Variables in TS

- Variables act as a container for value in code and must be declared before the use.
- In TypeScript, the variable follows the same naming rule as of JavaScript variable declaration.

1. The variable name must be an alphabet or numeric digits.
2. The variable name cannot start with digits.
3. The variable name cannot contain spaces and special character, except the underscore(_) and the dollar($) sign.

TypeScript

Variables in TS

- There are three ways of declaring variables in JavaScript:

1. var
2. let
3. Const

Difference between Let vs Var vs Const

var is function scoped

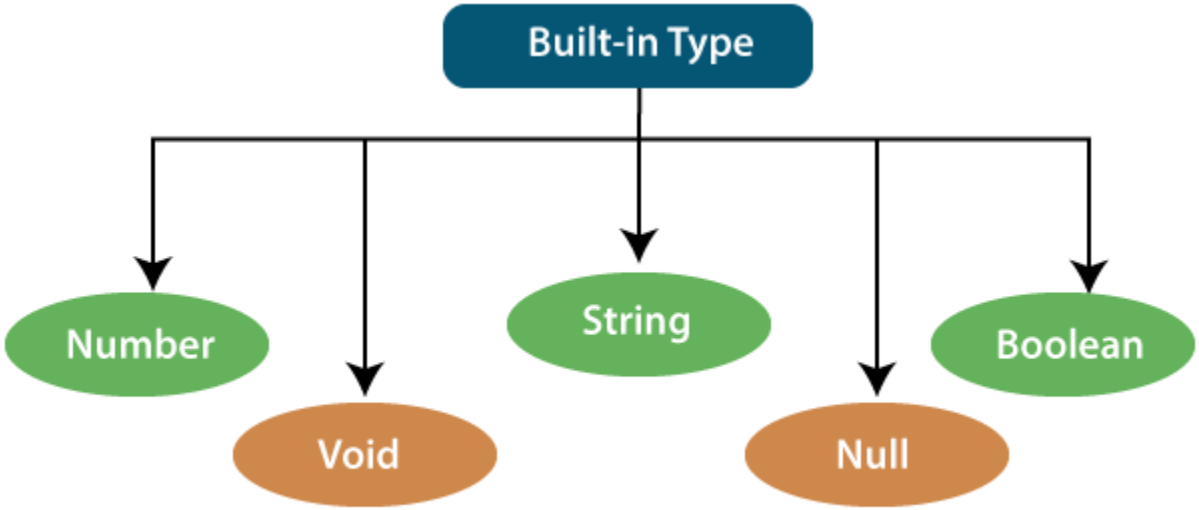
The variables declared using var inside the function are available only within that function. If we declare them outside the function, then they are available everywhere i.e. they are a global variable.

let & const is block scoped

The variables declared using let or const are block-scoped. They are scoped to the block in which they are declared i.e. inside the if/try/catch/while/for or any code block (enclosed in curly parentheses).

TypeScript

TypeScript Simple Types



TypeScript Simple Types

- TypeScript supports some simple types (primitives).

There are seven main primitives in JavaScript and TypeScript.

1. **boolean** - true or false values
2. **number** - whole numbers and floating point values
3. **string** - text values like "TypeScript Rocks“
4. **Null** - has one value: null
5. **Undefined** - has one value: undefined. It is a default value of an uninitialized variable
6. **Symbol** - represents a unique constant value (rarely use)
7. **Bigint** - used to represent and manipulate primitive bigint values — which are too large to be represented by the number primitive. E.g. 900719925474099 1n

TypeScript Simple Types

Type Assignment

- There are two main ways TypeScript assigns a type:
- **Explicit** - writing out the type

```
let newVar: string = "Haroon"
```

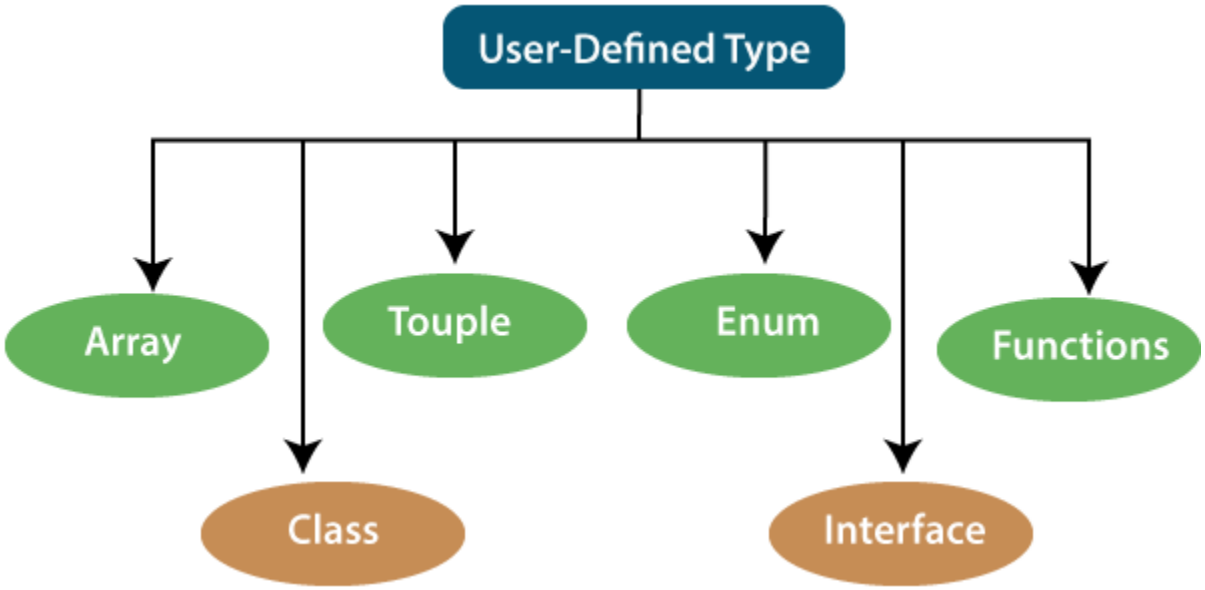
- **Implicit** - TypeScript will "guess" the type, based on the assigned value:

```
test.t  let newVar: string
let newVar = "Haroon"
```

TypeScript

Additional Types (Object Types)

Object types (User-Defined Type)



TypeScript

Additional Types (Object Types)

Object types

1. **Fonctions** - The type of the value returned by the function can be explicitly defined.
2. **Array** - An array is a special variable, which can hold more than one value:
3. **Tuples** - A tuple is an array with fixed size and known datatypes. we use a tuple for a static, well-defined array.
4. **Enum -** An enum is a special "class" that represents a group of constants (unchangeable) variables.
5. **Union -** TypeScript allows us to use more than one data type for a variable or a function parameter. This is called union type.
6. **Any** - any is a type that disables type checking and effectively allows all types to be used.
7. **Void** - A void is a return type of the functions which do not return any type of value. It is used where no data type is available. A variable of type void is not useful because we can only assign undefined or null to them.
8. **Never** - The never type is a type that contains no values. Because of this, you cannot assign any value to a variable with a never type.

Thank You