Typescript

Marina Magdy

Agenda

- Typescript overview
- Why using Typescript
- Installation steps
- Types
- Interfaces
- Classes
- Access Modifiers
- Modules

What is Typescript ?

Docs: https://www.typescriptlang.org/docs/handbook/intro.html

- TypeScript is an open-source, object-oriented language developed and maintained by Microsoft
- TypeScript extends JavaScript by adding data types, classes, and other object-oriented features with type-checking. It is a typed superset of JavaScript that compiles to plain JavaScript.

Why using Typescript !

- JavaScript is a dynamic programming language **with no type system**.
- JavaScript variables are declared using the var keyword, and it can point to any value. JavaScript doesn't support classes and other object-oriented features.
- The type system increases the code quality,readability and makes it easy to maintain and refactor codebase. More importantly, errors can be caught at compile time rather than at runtime.
- TypeScript compiles into simple JavaScript.

Typescript : Getting Started

TypeScript code is written in a file with .ts extension and then compiled into JavaScript using the TypeScript compiler. A TypeScript compiler needs to be installed on your platform. Once installed, the command tsc <filename>.ts compiles the TypeScript code into a plain JavaScript file. JavaScript files can then be included in the HTML and run on any browser.

- npm install -g typescript //install typescript
- tsc -v //check typescript version and make sure that it's installed
- tsc file.ts --watch //run command to compile ts file to js

Typescript : Types

- **Duck typing** is the ability of a language to determine types at runtime. JavaScript follows the duck typing paradigm; it has types but they are determined dynamically and developers don't declare them.
- TypeScript goes one step further and adds types to the language. Having data types has been proven to increase productivity and code quality, and reduce errors at runtime.
- It's better to catch errors at compile time rather than have programs fail at runtime.

Typescript : Types

- String
- Number
- Boolean
- Array
- Object
- Any
- Functions
- Union Types
- Interfaces

Typescript : Types

Example :

```
let projectStatus  = 1;

projectStatus = 'Success';


//Error: Type 'Success' is not assignable to type 'number'
```

Reason :

TypeScript determined the type as number and then alerted us when we were trying to assign a string.

Typescript : Interfaces

An interface is an abstract type. It only defines the 'signature' or shape of custom types. During transpilation, an interface will not generate any code, it is only used by Typescript for type checking during development.

Example :

```
interface Product {
  productName: string;
}
Let productItem : Product {
  productName:”Lipton”;
}
```

Typescript : Interfaces

```
user {  name , age , education , job , experience } -> type object ->
Type User

Custom Type for Users :

Interface User {
     Name : string;
     Age : number ;
     Education : Array<String> // [‘education1’ , ‘Education2’]
     Job : string; // ‘Job’
     Experience : Array<String>
}
```

Typescript : Classes

TypeScript is object oriented JavaScript. TypeScript supports object-oriented programming features like classes, interfaces, etc. A class in terms of OOP is a blueprint for creating objects. A class encapsulates data for the object.

A class can include the following:

- **Constructor :** method which is called when creating a new instance of class.
- **Properties :** is any variable declared in a class
- **Methods :** any functions represent actions an object can take.

Typescript : Constructors

The constructor is a special type of method which is called when creating an object. In TypeScript, the constructor method is always defined with the name "constructor".

If the class includes a parameterized constructor, then we can pass the values while creating the object.

```
class Product {
  productName: string;
  constructor() {}
}

Let product = new Product();
```

Typescript : Access modifiers

Access modifiers change the visibility of the properties and methods of a class.

TypeScript provides three access modifiers :

- **Private :** The private access modifier ensures that class members are visible only to that class and are not accessible outside the containing class.
- **Protected :** The protected modifier allows properties and methods of a class to be accessible within same class and within subclasses.
- **Public :** By default, all members of a class in TypeScript are public. All the public members can be accessed anywhere without any restrictions.
- **Readonly :** We can access read only member from the outside of a class, but its value cannot be changed.

Typescript : Modules

A module is a way to create a group of related variables, functions, classes, and interfaces, etc. It executes in the local scope, not in the global scope. In other words, the variables, functions, classes, and interfaces declared in a module **cannot** be accessible outside the module directly. We can create a module by using the **export** keyword and can use in other modules by using the **import** keyword.

File1.ts : export var greeting : string = "Hello World!";

File2.ts

Import { greeting } from file1 ;

console.log(greeting);

Thank you

Task 1

- Install Typescript and setup the development environment.
- Create a class called vehicle that has color - string- and type - string -
- Create another class called Car that extends vehicle class and has price - number -
- Create a new instance from car class and with (color : ‘black’ , type : ‘PMW’ , price : “1,000,000”)
- Create another instance from car class with (color:”red” , type : “Kia” , price: “300,000”)
- Alert the output value each time you create a new instance.

Task 2

- Create a counter class that has counter value - number
- Counter Class contains 3 functions one to increase the counter value , one to decrease the counter value and the last one to reset counter value
- Create a new instance from Counter class and use the class inner functions to change the counter value.
- Append the the counter value to the html.