Typescript Short Tutorial

Based on **Essential TypeScript 4**
By Adam Freeman

Why Typescript

- Javascript is must have but very unstructured, dynamic and difficult to manage for enterprise applications. Typescript is a solution for such issues.
- Web Assembly is the future, a major platform for the web. Node.js and JavaScript is a perfect match with Web Assembly. AssemblyScript is a TypeScript like language for Web Assembly. TypeScript + Assembly Script is the future for both client and server programming.
- Ethereum and other digital currency are also moving to webassembly, and AssemblyScript will help us write Smart Contracts.
- Typescript help us learn OOP paradigm very well. Could help you to learn other languages like Go, Python, C# , and Java.
- Due to well defined structure, you can enjoy many IDE support like Intellisense and error identification on the fly

Creating a Todo List in typescript (Hands on)

We will start coding in small iterations. Start from basic and go the to complex

Creating a Todo List:

the Model (Represent single object ⇒ data (attributes) and method(operaitons) related to a single task

```
export class TodoItem {
    public id: number;
    public task: string;
    public complete: boolean = false;

    public constructor(id: number, task: string, complete: boolean = false) {
        this.id = id;
        this.task = task;
        this.complete = complete;
    }

    public printDetails() : void {
        console.log(`${this.id}\t${this.task} ${this.complete
            ? "\t(complete)": ""}`);
    }
}
```

Class is the key component in typescript
It has attributes (id, task, complete) that holds data
And method that are used to perform operations(printDetails). This is very simple class. Will complex later on

constructor is a method that runs anytime when we call the class

Note: Only for explanation, Use the book code and repo for copy paste and run codes:

Creating a Todo List:

```
import {TodoItem} from "./TodoItem";

export class TodoCollection{

    private nextId: number = 1;

    constructor(public items: TodoItem[] = [] ){

    }

    public addTodo(item: string): num
        this.items.push(new TodoItem(this.ne
        return this.nextId++;
    }

    public printAll(): void {
        this.items.forEach((item) => item.printDetails(
    }

    public getItem(id: number): TodoItem {
        return this.items.find((item) => item.id == id);
    }

    public taskDone(id: number): void{
        this.getItem(id).complete = true;
    }
}
```

Collection means set of related objects

Every method provide details of operations to be performed

Creating a Todo List: the index.ts

```ts
import { TodoItem } from "./todoItem";
import {TodoCollection} from "./todoCollection";

let list : TodoCollection = new TodoCollection();

list.addTodo("Buy Mango");
list.addTodo("Buy Meat");
list.addTodo("Get Haircut");

list.taskDone(2);

list.printAll();
```

```
/FETypeScript/todo$ tsc
src/todoCollection.ts:1:24 - error TS2307: Cannot find module './TodoItem' or its corresponding type declarations.

1 import {TodoItem} from "./TodoItem";


Found 1 error.
```

Creating a Todo List: Testing, debugging and Running

```
/FETypeScript/todo$ tsc
src/todoCollection.ts:1:24 - error TS2307: Cannot find module './TodoItem' or its corresponding type
declarations.
1 import {TodoItem} from "./TodoItem";

                ~~~~~~~~~~~~~

Found 1 error.

FETypeScript/todo$ tsc

FETypeScript/todo$ ls dist/
index.js          todoCollection.js  todoItem.js

FETypeScript/todo$ node dist/index.js
1       Buy Mango
2       Buy Meat        (complete)
```

The tsc command is used to trasnspite typescript into javascript. And could also identify error

No error: means transpilation successful

Executing using node