Angular by Mosh

# 003 Architecture of Angular Apps



* HTTP Services / APIs
  + Endpoints (server port) that are accessible via the HTTP protocol
* NPM
* Angular cli
  + Sudo npm install -g @angular/cli
  + ng --version
  + **npm start** 🡪 **ng serve**

# 006 Structure of Angular Projects

* E2e – end-to-end test. This is automated tests that simulate a real user
* Node\_modules – where all the third-party libraries are stored.
  + We *don’t* deploy node modules to a server
  + Use 🡪 **npm install** 🡪 to install necessary package for your app.
* Src – source folder
* App – the folder contains a module and components
* Assets – the folder to store the static assets of our application: img, text, icon, etc.
* Environment – the folder to store the configuration settings for different environment.
  + Production environment
  + Development environment
* Main.ts – bootstrapping the main module of our application
  + 
  + Everything starts from loading the AppModule
* Style.css – global styles
* .gitignore – exclude certain files and folders from your git repository
* Tslint.json – settings for TS linked. TS lint is a static analysis tool for type script code for readability, maintainability and functionality errors.

# 007 Webpack

* Hot Module Replacement (HMR) – reloading web browser automatically after one of the source files is modified and saved.

# 013 What is TypeScript

* Super set of JavaScript
* Strong typing (optional) – make it more predictable
* Object-oriented features
* Compile-time errors – catch error at compile time instead of run time
* Great tooling
* Compile or transpile TypeScript code into JavaScript code that browser can understand

# 014 Your First TypeScript Program

* sudo npm install -g typescript
* To transpile TypeScript into JavaScript 🡪 tsc filename.ts 🡪 filename.js
* To execute the code 🡪 node main.js/main.ts

# 015 Declaring Variables

* JavaScript versions: ES5, ES6 (2015), ES2016, ES2017, …
* var -vs- let

function doSomething() {

for(var i = 0; i < 5; i++) {

console.log(i);

}

console.log('Finally: ' + i);

}

doSomething();



* change to let

for(let i = 0; i < 5; i++) {

* Although we have a compilation error, TypeScript compiler still generated main.js
  + Typescript compiler reports the errors but still generate **valid** JavaScript code

# 016 Types

* In .ts
  + Use 🡪 let
  + Changing the assigned type will cause compilation error but still generates valid JS
* In .js
  + Type is converted to 🡪 var
* Any
  + Let variableName;
* Type annotation

let a: number;

let b: boolean;

let c: string;

let d: any;

let e: number[];

let f: any[] = [1, true, 'a', false];

* Enum

enum Colour { Red = 0, Green = 1, Blue = 2};

let backgroundColor = Colour.Green;

# 017 Type Assertions

* Intellisense in VS code
* Need to explicitly tell typescript compiler that this message variable is actually a string (any types) – type assertion

let message;

message = 'abc';

let endWithC = (<string>message).endsWith('c');

let alternativeWay = (message as string).endsWith('c');

# 018 Arrow Functions

let doLog = (message) => console.log(message);

# 019 Interfaces

* If you have a function with **many** parameters, you can pass a **point object**.
* In-line annotation: 
* Interface will solve this redundancy problem.
* Pascal naming convention for interface name is capitalizing the first letter