1. Installing Typescript:
   1. Go to <http://www.typescriptlang.org/>, click “get type script now”, select npm (Npm (Node.js Install pack. manager), in case with VS we should be use in NuGet.)
   2. Go to <https://www.npmjs.com/>, find “installinh npm”
   3. After installing npm we can install typpescript – type in cmd : **npm install –g typescript,** then type: **tsc –v**
   4. Installing tsd(TypeScript definition manager):
      1. Before this: **TypecScript definition files:** (.d.ts) not deploeyd – this files describes the type defined in external libraries. We need this to external libraries and framworks that we use, the compiler uses this file for static type checking. We have community driven repository of .d.ts [DefinitelyTyped](http://definitelytyped.org/). Go to repository.
      2. DefinitelyTyped provides the tool that helps provide and install .d.ts files, it’s called TypeScript Definition Manager (tsd) – locates and install .d.ts files directly from the [DefinitelyTyped](http://definitelytyped.org/).
      3. To install type in cmd: **npm install tsd –g**
   5. Summary: Install nmp, install typeScript, install tsd.
2. Setting up vs code for Typescript:
   1. Define a folder: create folder outside from VS code
   2. Open folder with VS code
   3. Create configuration file - tsconfig.json: here we should to configure the complier options
   4. Configure task runner – tasks.json – used by typescript compiler
3. Run code:
   1. Install http-server open cmd and type: **npm install –g http-server**
   2. Run it: type in cmd **http:server**
4. Angular modules:
   1. One main module – **angular.module(“productManagement”,[]);**. Here is we need to .d.ts files to start working on .ts with angular, ope cmd in the folder and type: **tsd install angular --resolve – save**. The .t.ds files should be added to the folder called **typings**.
5. Creating an Angular Controller in TypeScript
   1. Interface for our intent: interface defines the set of methods and properties that we intent to expose to the view.
   2. Class for the controller
   3. Registration with an Angular module
6. TypeScript Modules:
   1. Encapsulate variables, interfaces and classes:
      1. Interfaces and classes within module are local to the module.
      2. Expose the interface or class with keyword **export**
   2. Define unique namespaces (system.data, java.io …) **module *your\_namespace* {…..}**
   3. Provides two types:
      1. Internal
      2. External
   4. TypeScript module conventions:
      1. Put iffy around each module and controller
      2. Define namespace for module:
         1. App – root namespace
         2. Define sub-namespaces for each feature, module, domain classes, common code
7. To use a custom service:
   1. Identify the module dependency
   2. Inject the service into the controler constructor func.