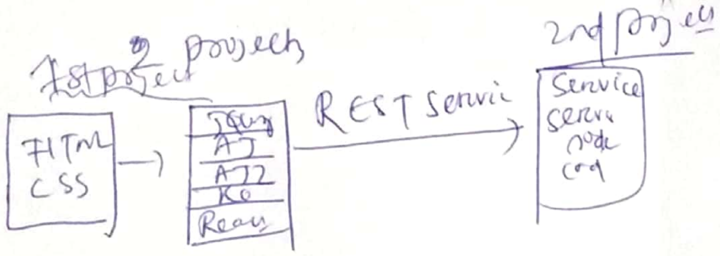
$$\begin{tikzcd}
 & \text{ES6} \arrow[d] & \\
 & \text{Object Oriented JS} \arrow[dl] \arrow[dr] & \\
\text{JS} & \text{TypeScript (MS)} \arrow[l, "\text{Transpile}"'] & \text{Dart (Google)} \arrow[r] & \text{for mobile app \& all}
\end{tikzcd}$$

Client Side Editor

1. ATOM
2. WebStrom
3. ~~[illegible]~~ Sublime
4. Vistual Studio Code (MS)
   (code.visualstudio.com)
   Step 1 → install for windows

Jquery :- is a library.

Angular - is a F/W. (collection of library)



Node JS :- is a repo for diff js libraries which allow JS ~~to~~ be executed in server side.

npm → node package mgr (like nuget pck mgr)

② NpmJs.com (STEP2)

→ Scrolldown & download window installer 64bit & install

→ Open node.js command prompt in Start menu

→ npm install jQuery
" " bootstrap } then
" " -g @angular/cli → once this cmd runs succesfully then all ng command can be fired like ng new Day1

↳ just like GAC (all project inside root folder can access all the libraries)

the libraries will be available for root folder where u install this.

Once done in root folder path try below

→ ng new Day1

this will create a AJ2 project structure.

→ Once you install ~~ao~~ angular cli chto node cmd prompt then u will's able to install ng ~~comman~~ command ch node cmd prompt.

package.json ÷ Kind of webconfig. Contains all dependancies.

In ~~Cmd~~ node cmd go to Day 1 folder & install Jquery.

npm install jquery --save

Now u can see jquery line added in pckg.json.

rxJs ÷ reactive extension → for ajax call angular uses this.

<u>Dev.Dependacies</u>

tslint ÷ define the rules of ; or space allowed or not.
Check the rule ch tslint.json file.

.spec.ts ÷ used for unit ~~testing cases~~.

→ Fire command ng serve
this will compile the application & final msg "webpack: compiled successfully".
type localhost : 4200 ch browser → u will see dummy app.

<u>To change port number.</u>

Ctrl+C → Y ch cmd to terminate running app.

Day 1 ng serve --port portnumber → 4 digit 1111

now ch browse ÷ localhost : 1111

Bundling is done by webpack, rendering of bundle is done by systemjs

Component

If you write single line inside template then use sing quote.
if multiple line then used back tick (above tab key)

Module is a namespace in C#, Component is class in C#

Create app.component

Create app.module & mention the component dependancy.

Open a file : .angular-cli.json
this file contains start up Html file & ts file. U can change.
(Html file → index; ts file → main.ts)

HMR → Hot module Replacement.
change a value in component (Welcome Angular 2 → 4)
and save, on browser U will see new Value. called HMR

$$\begin{tikzcd}
\text{index.html} \arrow[r] & \text{main.ts} \arrow[r] & \text{app.module} \arrow[r] & \text{app.component} \arrow[lll, bend left]
\end{tikzcd}$$

<form #form="ngForm"> → id of form in angular 2
ngForm is angular 2 enabled.

name="name" ngModel
↳ by mentioning this the pristine will be false once U type anything in textbox.

in angular 2 "novalidate" → disable html validation.
" 4 by default. it comes.

at form tag mention ~~ngNo~~ ngNativeValidate to enable Html validation so that U can write "required" in elements
↳ html element.

To get the bootstrap online:-

browse: getbootstrap.com.

~~rightcorner~~ top right corner, Choose version 3.7.7.

Click on getting started in top menu.

copy first link of CDN in that page & paste in head section of index.html.

anything mention inside [ ] in one way binding.

ex: [ngModel]

{{form.value | json}} – print value of all controls on the page. inside a form.

2 way binding: [( )]

[ ] → ~~event~~ oneway binding

[( )] → 2waybinding

( ) → event binding (all methods are written inside)

Ex: (ngModelChange) → the best method for textbox change event.

<input type="text" #firstname class="form-control">

{{firstname.className}} → return set of classes.

ng touch: works if you lost focus on control.

pattern="...+" maxlength="8"

↳ minimum 3 char

① Bootstrap appl^n with~~out~~ module. ✓

② **Directives** (Used to reference ~~a~~ component selector in other component template)

directives ~~@~~: [HeroListComponent]

↗

this is used because if you mention selector of component inside the template of other component. Then is to get the reference of that selector.

Ex

```
                 → app component
@ Component ({
      selector : "superhero-app',
      template: '<div>"
                 <h1> {{Item}} </h1>
                 <super-heroes> </super-heroes>
                 </div>',
      directives: [HeroListComponent]
```

selector defined inside HeroListComponent.

~~@~~index.html
↳ Main.ts → AppComponent
⇓
HeroListComponent

Day 2

let : like dynamic in C#
any : like var " "

✓③ **Pipes** : uppercase, lower case, number, currency.

⇨ In the beginning of day 2 exercise for ~~@~~Indian Super hero, type npm start command to run application.

Hide js & .map file from sol^n Explorer :
File → preferences → settings : Workspace setting (Right ~~top~~ corner)

add

```
"**/*.js": true,
"**/*.map": true
```

④ ~~P~~ **Life Cycle Events**

1) ~~Onboard~~ On Init
2) On Change.
3) On Destroy

✓⑤ **Service / Providers**.

http.dev.js → index

app.ts : import {HTTP_PROVIDERS} from "angular 2/http";
import 'rxjs/Rx';
provider: [HTTP-PROVIDER].

Day3

Create & Consume a http service

$.ajax (url, success, failure) → ajax call in JQuery. In the middle if U want to cancel ~~it~~ they, not possible

success & failure methods are called "Promise"

(1) can not cancel ops.

(2) 2 diff api called parallely

(3) If search by same key then not call again & again (cache concept).

(4) Each type of character go to server & get data like 's' typed, 'a', 'c', 'h', 'i', 'n' .. for each type of char go to server + get data

These 4 are not possible in jquery here we have :-

rxjs → Reactive extension.

↳ It gets observable not promise

(*) Use of arrow function. ⇒ in Angular js 2.

- map( ) → returns data
- do ( ) → do any kind of logic
- catch(method) → calls the method to handle error.

→ Observable always comes with subscribe. Caller of Observable uses subscribe.

To implement httpservice :- (1) add http.dev.js file ref in script tag in index.html.
(2) In app.component.ts :- import {HTTP_PROVIDERS} in top & inside @Component add providers: [HTTP_PRO..]

Enable you api with CORS for the consumer.

Routing

In index.html add script ref to router.dev.js. (2) Add line <base href="/" />
(3) In app component add import {ROUTER_PROVIDERS, RouteConfig, ROUTER_DIRECTIVES} from angular/router
anchor tag will be wrapped by Router_DIRECTIVES.

[RouterLink] = "['Welcome']"
↳ This must be capital always.

(4) RouterConfig ([{path: "welcome", name: "Welcome", component - ..