5:30:29

To create a new Angular project in Angular5 .

**Insallations:**

1.Latest node :https://nodejs.org/en/download (node -v ->gives version)

2. Latest angular CLI :npm install –g @angular/cli (ng -v ->gives cli version)

3. Install visual studio code.

1. Go to application folder
2. Ng new <application name> -style=scss –routing (scss for saas, routing to integrate and set up routing by default.) To skip test files use –skip-tests true
3. Go inside the new folder and run **ng serve**. Ng serve –o will build and launch the application in default browser.
4. If we get an error like cannot find @angular-devkit/core , then do the following steps
   1. Npm update –g @angular/cli
   2. Edit package.json line “@angular/cli”:”1.6.0” to “@angular/cli”:”^1.6.0”
   3. Npm update . This will resolve error by installing latest version
   4. To install animations >npm I @angular/animations
   5. To insall typescript latest version >npm install typescript@latest - -save-dev
   6. To update core >npm update @angular/core @latest
   7. To install bootstrap, npm install bootstrap@3 –save. To include in the project , add it in angular-cli.json ->styles ->../node\_modules/bootstrap/dist/css/bootstrap.min.css
5. After creating project go inside project folder and type code. This opens the code in visual studio code .(code space dot).

**Clone project:**

Git clone repository\_url ,

npm install (this resolves all dependencies).

**Create Component:**

1. Ng generate component <component\_name> will generate .html,.ts and .scss and .spec.ts. also it updates app.module.ts.
2. Ng g c <component\_name> is a shorthand form.
3. To create inside a folder , give ng g c foldername/componentname. This will create the folder inside src/app folder in the project. **--spec false** to avoid creating spec. **--flat true** is used to avoid creating separate folder for each component.

So ng g c employees/listEmployees –spec false --flat true.

OR ng g c employees\listEmployees --flat --no-spec

When code is changed, page is refreshed automatically even before saving the changes.

Global styles can be defined in src/styles.scss. Kendo styles are imported there.

Import styles from url using @import url(‘url’);

**Property Binding**

Either we can write value = “{{propertyName}}” OR [value]=”propertyName”.- One way :component to template.

**Two way databinding**

This is achieved using ngModel.to use it need to include forms module in app.module.ts.

It is specified like [(ngModel)]=”propertyName”

**Event Binding**

In Html element , (click) = “function name”

**Array:** To declare array , goals = []; to add item to array, this.goals.push(item); To remove item from an index , this.goals.splice(i , 1);

**Animation:**

To use animation, npm install @angular/animations@latest –save. (inside project folder)

Then import BrowserAnimationsModule from @angular/platform-browser/animations in app.module.ts and add to imports array inside that.

Then import specific functionalities in the respective .ts file.

We can define animations:[] inside @component.

**Update dependencies to latest**

Go to the project folder and run npm update

**Routing**

App-routing.module.ts – contain routing information.

Import {HomeComponent} from ‘./home/home.component’;

Routes –contain route collection where we can give path and component and redirectTo.

To receive paramaters from url import ActivatedRoute from @angular/router inside the specific component. Then need to create an instance in constructor. Then subscribe this.route.params.subscribe();

**Component Based Router Navigation**

Route based on logic inside a component class. Need to import Router from angular/router in the component. Then create an instance in constructor say *router*  and sat this.router.navigate([‘route as per routing Module’]);

**Services**

Helps to communicate between components. Share data between components or make http calls.

To create a service, ng generate service serviceName OR ng g s serviceName.

Best ways to communicate using BehaviorSubject in RxJS. So import BehaviorSubject from ‘rxjs/ BehaviorSubject’.

Need to define observable to share data. Eg: goal = this.goals.asObservable();

Register the service in app.module.ts : import {Service} from ./servicename.service and add it to provider array.

Then import the service in the component.create an instance in constructor. Then in ngOnInit, this.instanceName.ObservableInsideService.subscribe(res=>this.variableName = res);

Also we can call service method here.

**Deployment**

Run ng build command. This will create a dist folder inside project folder.

For deploying to other servers, run ng build –prod. This will make significant modifications to make files smaller and faster.

If uploading to a subfolder like yoursite.com/, build using ng build --prod --base-href=”url”

Another way is to use github pages.npm install -g angular-cli-ghpages.

For this need to install git., Create an account in github , create repository using the account.

Then use command line and go to project folder.

>git add . (adds all modified and new files in current dir and subdir to the staging and thus preparing them to be included in the next git commit.

>Git commit –m “first commit”

>git remote add origin <https://github.com/vidyaqph/ng5.git> (this is copied from the repository instructions

>git push –u origin master ( this is also copied from instructions)

>ng build --prod –base-href=”<https://vidyaqph.github.io/ng5>”

>angular-cli-ghpages

This will publish the app in the url <https://vidyaqph.github.io/ng5>

If we are deploying the code to a subfolder , then index.html should contain <base href =”/subfolder/” . So either maulally edit it or give base-href /subfolder/ along with ng build.(ng build -prod --base-href /emp/

**Word wrap**: Visual studio code->view toggle wordmap.

**Routing:**

Import RouterModule,Routes from angular/router (in app.module.ts) .then can define routes as an array of Routes. In the imports array, need to add RouterModule.forRoot(routeVariable). Then we can define the navigation in app.component.htlml(for displaying on all pages. Also need to include <router-outlet> tag.

**Create Forms**

2ways to create forms in angular.

1)Template driven forms- used to create simple forms.

2)Model driven forms/Reactive Form.- more complex.

Whenever angular see a form element it attaches ngForm

**Label**

If we use for=”inputFieldName” on label, when we click on label associated input element receives control

**Radio Button**

To create radio button, <input type=”radio” value=”female” name=”gender”> inorder to make multiple radio buttons mutually exclusive, provide same name for all the radio buttons like name = “gender”.

**Note:** Checked attribute of radio button does not work with ngModel. Instead need to create a property in component class and set default.

To disable a radio button/checkbox just use disabled .By default , disabled controls re not included in angular auto generated form model.

**Button**

When we don’t set the type property on button, it behaves as a submit button. Need to set it as type=button inorder to change this behavior.

**Validation**

By default , in Angular 4 and later , browser validation is disabled . by default, it generates **novalidate** attribute on form element .

**ngNativeValidate** directive will enable default browser validation on html elements.

Note:Different browser vendors implement browser validations differently , so users will get different experiences. So it is better to avoid that and do validations in angular.

**IE:** Inorder make the application work in IE, need to uncomment code in polyfill.ts which are related to ie. OR import core-js

**Form Validation in Angular**

Angular uses html validation properties to validate control data.inorder to do this use local variable or template variable (or called local template reference variable). Either we can define a model class and give that model or angular auto generated model .if using autogenerated, need to define a local variable and assign ngModel directive and use it to check the below validation properties.

Eg: <input type=”text” id=fullName” required name=”fullName”[(ngModel)] = “fullName” #fullNameControl = “ngModel” where fullNameControl is the template variable.

We can use{{ fullNameControl.touched}}.

So, 1) Include html5 validation attrbutes such as required.

2)Export ngModel Directive to a local template variable.

3) use this local template variable to access validation properties.

Angular Form validation properties:

Touched – field is touched (even when we click on the field and tab it will be true)

Untouched – field is not touched.

Pristine – field is not changed

Dirty- field is changed

Valid - valid

Invalid – invalid

**Display Validation error messages**

Bootstrap classes: has-error, control-label,help-block

Use class binding: [class.has-error] = “fullNameControl.invalid && fullNameControl.touched ” fullNameControl..touched ensures that the error is not displayed on initial form load.

Pattern validation can be used for complex validation.

To validate email either use html validator email or (email validator validates empty string. So need to condition [email] =”employee.email!=’’” inorder to exclude empty string.

**Angular safe navigation** : ? iscalled angular safe navigation and it protects against null and undefined values. It protects against null reference exception.

Built in angular email validator has a bug and it checked whether null or valid email and gets fired. So inorder to work around , need to give an initial value to email property to ‘’ and bind a condition to email validator like [email]=”employee.email!=’’”

eg: element id.errors?.required will check whether any required field error is there.

ngValue vs value : if the option ( for a dropdown )value is a string use value property, and if the option contain null value , use ngValue.we can use ngValue to pass object. If we give ngValue as an object name, entire object will be passed.

## Custom Validator

To create a custom validator , create a directive and use it on the element. Can use the selector for the directive on html elements. Need to register the directive against NG\_VALIDATORS (opaque token). Need to register it in the app.module. to pass default value to validate from html element to directive , use @Input . pass value like appSelectValidator = “select” from html. Inside directive define @Input()appSelectValidator:string and compare. To use an alias name, use @Input(‘appSelectValidator’) defaultValue :string so that we can compare defaultName.

**===** checks value and underlying datatype.

**Tslint.json**  contains all rules. We can change lint rules here.

## ngModelGroup

directive is used to group elements together. Eg: to display error message on both password and confirmPassword fields when they do not match put them together inside a div with ngmodelgroup directive

Trigger validation manually.

use **updateValueAndValidity**() function to trigger a form control’s validation. <controlName>.control.updateValueAndValidity() will fire all validations on the control.

Another way is to give the directive on ngModelGroup div so that validation happens whenever the controls inside the group changes. Then need to change the validation group accordingly.

## EVENTS on Controls

Change event: fires only after the typing is completed and control looses focus.

Input event: fires when the user types in an input field.

## Services

Service is a class.

Irrespective of whether a service has an injected dependency, angular recommends to use @injectable decorator.

If a service is registered at component level , it is available only to the component and its children.

To use a service in a component , inject it into the component constructor.

## Switch Case

**ngSwitch**

**:** [ngSwitch]=”variable to check”

<span \*ngSwitchCase = “value”>display name</span>

ngSwitchDefault is used to give a default value.

**ngSwitchCase:**

**ngSwitchDefault:**

## Create a child Component

1. Create a component normally using ng g c employees\listEmployees --flat --no-spec
2. Put all html containing model variable in child component and in .ts create @Input() modelVariable:Type ;
3. Add the decorator <child selector [modelVariable]=”passedVariable”> in parent component.

## Detect and react to Input property changes

1. One way is to use property setter
2. Another way is to use ngOnChanges lifecycle hook.

### ngOnChanges

ngOnChanges(changes:SimpleChanges) (take care of correct spelling.)

Chnages contain previous value and current value.

Useful when multiple properties change.we get all changes instead of just one single change.

### Property setter

Specific to a single property and gets the changes of only that property.

@Input() property : used to pass, data from parent to child component.

@Output() - Used to pass data from child to parent as EventEmitter.

**@Input()** emp :number. (specified in child)

In the child component,

**@Output()** notify : EventEmitter<string> = new EventEmitter<string>();

Pass data as this.notify.emit(value) , where value is event data which is termed as event **payload.**

Now, parent need to handle the event. Parent element specify (notify) = “handleNotify($event)” where $event will receive the payload.

Child component passes data to parent using event.

**Another way to call child component public methods and properties from parent:**

Template reference variable.

Create a template refrence variable like #childComponent on the parent component’s child component element and can call child property or method like childComponent.emp.name

## Routeguards

**Why they are required? :**Suppose we are on an edit form and we accidently click on a link .it immediately navigate there and lose data. It is good to keep a confirmation whether to navigate before do actual navigation. It can easily be achieved using CanDeactivate Routeguard.

CanDeactivate Routeguard – guard navigation away from current route.

CanActivate – Guard navigation to a route.

CanActivateChild – guard navigation to a child route.

CanLoad – guard navigation to a feature module loaded asynchronously.

Resolve – perform route data retrieval before route navigation.

**3 steps t build a route guard**

1. Build route guard –Create a service which implements CanDeactivate<componentName>. Then inside the CanDeactivate method check whether the form is dirty. Need to define @**ViewChild**('employeeForm') public createEmployeeForm: NgForm;//This is used to reference this component to check whether form is dirty or not in CanDeactivateGuard
2. Register the guard with angular dependency injection system – import the service in app.module(or respective module) and define in providers.
3. Tie the guard to a route-.- provide canDeactive in appRoutes(inside app.module or respective module).Provide like {path: 'model', component: CreateEmployeeModelComponent , canDeactivate: [CreateEmployeeCanDeactivateGuardService] }

If you want to access following inside the Parent Component, use **@ViewChild** decorator of Angular.

1. Child Component
2. Directive
3. DOM Element

ViewChild returns the first element that matches the selector.

Any element which is located inside the template, is **ContentChild**.

**Note: canDeactivate** does not work if we type the url or close the browser.

## Pass parameters In Route

Define route like { path: 'employees/:id', component: EmployeeDetailsComponent} in appRoutes inside module.ts. to pass multiple paramters , { path: 'employees/:id/:gender'…

Then need to inject route to constructor of the class from where the navigation to be done. Use this.\_router.navigate(['./employees', empId]);

Or we can do declaratively like <a [routerLink] = “[‘/employees’,2”></a>

## Read Parameter value from route

To read parameter value from route , need to import ActivatedRoute from angular/router. Also need to inject it in constructor.

Now either we can use snapshot approach or Observable approach.

### Snapshot Approach

This is good in scenarios like , when we want to move from one page to another to see some details.

Use this if the route parameter value does not change and you only need to read initial route parameter value.

Now get it like const id = +this.\_route.snapshot.paramMap.get('id');

OR

const id = +this.\_route.snapshot.params['id'];// Gets parameter id from route. + converts string to number.

**Note: params is deprecated since angular 4 and use paramMaps instead.**

### Observable approach

How to subscribe to angular parameter changes?

This approach is good when we want to navigate from one employee to another. Use this if the route parameter value changes and if you want to react and execute some code in response to that change.

Define like this.

this.\_route.paramMap.subscribe(params => {

this.\_id = +params.get('id') ;

this.employee = this.\_employeeService.getEmployeeDetails(this.\_id);

});

**Note:**If we use snapshot approach in this scenario, the route value changes ,but component gets initialized only once and will not be changing the employee values displayed on this.

## Convert to int from string easily

+ will convert string value to int.

Const a = +b will convert string b to int.

## Bootstrap Footer

To include a footer , just place a div with style class =”panel-footer”

Bootstrap right

Class = “pull-right” will display element to the right.

## routerLink

square brackets are not needed when we bind to a simple string like <a routerLink = “/list”>

square brackets are required if we bind to a property within component class or binding to a parameterized list like <a [routerLink] =”[‘/list’,id]>

## VS Code Editor

Format On Save –to auto format on save.

<https://angular.io/resources> contain resources and all 3rd party components.

Check Core version : dotnet --info