ANGULAR 7 PPT

N.H.L.Bhavani
ID:5067680

OBJECTIVE:

- Features of Angular 7
- Animation
- Material
- Guards

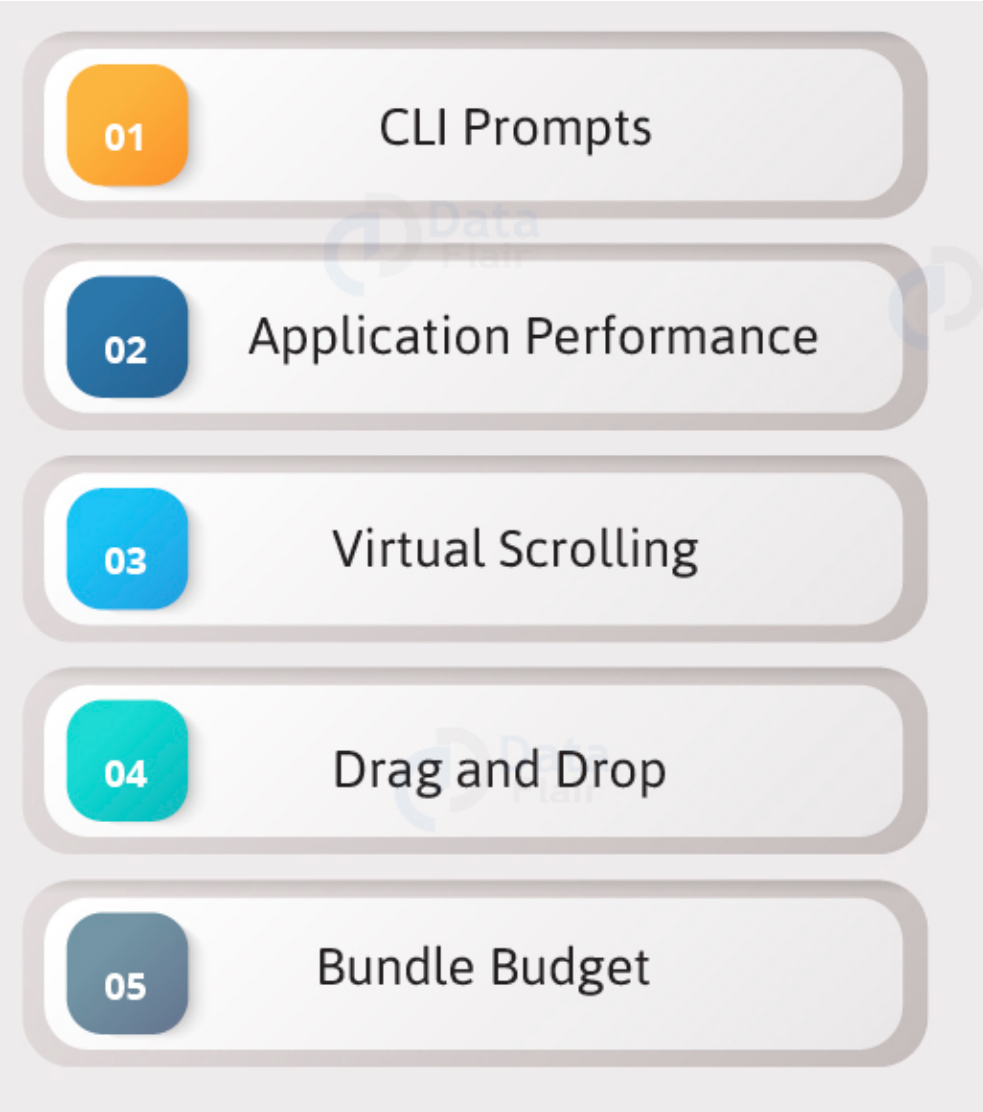
What is Angular 7?

- Angular is a framework to build a web application, which is becoming popular because of its unique features and ease to build an application. Angular 7 is an open source framework developed by Google. It completely relies on HTML and JavaScript. It converts a static HTML page into dynamic HTML page.

Prerequisites of Angular7.

- Angular2
- TypeScript
- HTML
- CSS

Features of Angular7…



1. CLI Prompts

It helps users to make a decision. It asks users about “want to add routing? Y/N” and about the type of styles user want to use.

Commads used in Angular7 projects…

Sr.No	Commands and Description
1	**Component** ng g component new-component
2	**Directive** ng g directive new-directive
3	**Pipe** ng g pipe new-pipe
4	**Service** ng g service new-service
5	**Module** ng g module my-module
6	**Test** ng test
7	**Build** ng build --configuration=production // for production environment ng build --configuration=staging // for stating environment

2. Application Performance

- Earlier reflect-metadata is used in production but it is required at the time of development. Therefore, ployfill.ts is removed by default in angular 7.

3. Virtual Scrolling

- Google accelerates the speed of Angular 7 for a huge scrollable list.

Start with a plain Angular table

- We’re starting with a pretty simple example of a table using a ***ngFor** loop

```
<table>
  <thead>
    <tr>
      <td>Name</td>
      <td>ID</td>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let row of tableData">
      <td>{{row.name}}</td>
      <td>{{row.id}}</td>
    </tr>
  </tbody>
</table>
```

The **tableData** property is defined in the corresponding component.ts file.

```
// app.component.ts
import { Component, OnInit} from '@angular/core';

@Component({
  selector: 'my-app',
  templateUrl: './app.component.html',
  styleUrls: [ './app.component.css' ]
})
export class AppComponent implements OnInit  {
  name = 'Angular';

  tableData = [];

  ngOnInit() {
    for (let i = 0; i &lt; 1000; i++) {
      this.tableData.push({
        name: 'Name' + i,
        id: i
      });
    }
  }
```

Introducing virtual scrolling

- The Angular CDK provides a scrolling component. We're now going to add it to our plain table in 4 simple steps.

1. Add the dependency

```
npm install @angular/cdk --save

```

2. Add ScrollingModule

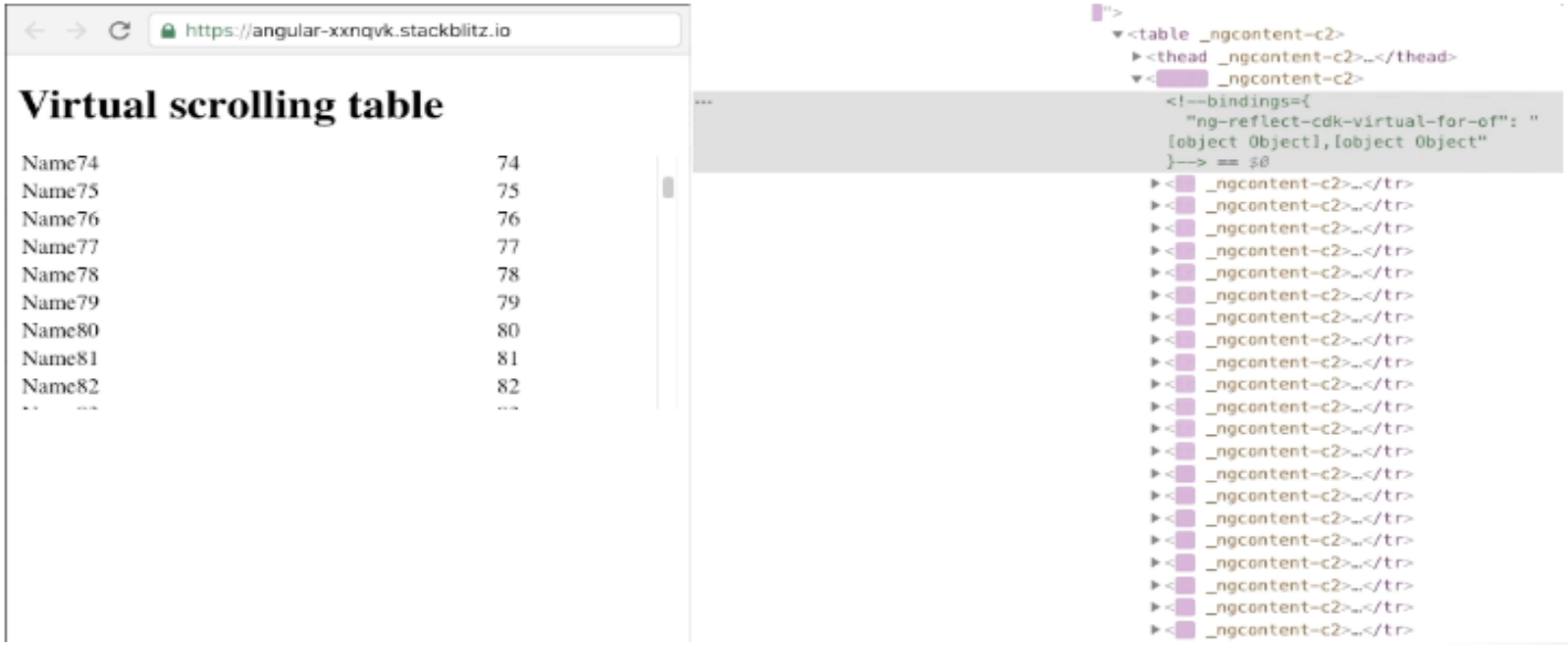
```
//app.module.ts
import { ScrollingModule } from '@angular/cdk/scrolling';

@NgModule({
  imports:      [ ScrollingModule ],
})
export class AppModule { }

```

- **3. Add Scrolling Component**
- Step 2 is to add the **<cdk-virtual-scroll-viewport>** element around the markup of your table. We need to provide the attribute **[itemSize]="heightOfRowInPx"** that tells the scrolling component how high each row is.
- **4. replace *ngFor with *cdkVirtualFor**
- instead of using ***ngFor** we're going to use ***cdkVirtualFor** that is needed in order for the virtual scrolling to work as intended.
-

```
<cdk-virtual-scroll-viewport [itemSize]="20">
  <table>
    <thead>
      <tr>
        <td>Name</td>
        <td>ID</td>
      </tr>
    </thead>
    <tbody>
      <tr *cdkVirtualFor="let row of tableData">
        <td>{{row.name}}</td>
        <td>{{row.id}}</td>
      </tr>
    </tbody>
  </table>
</cdk-virtual-scroll-viewport>
```

Result..

- If we inspect the DOM changes after introducing the **<cdk-virtual-scroll-viewport>** we see that the browser is removing and adding DOM Nodes as we are scrolling.



4. Drag and Drop

It comes with the feature of automatic rendering.

5.Bundle Budget

If the bundle size is more than 2MB, a warning message provided and for above 5MB, an error will be given.

Animation

Animations add a lot of interaction between the html elements. Animation was available with Angular 2, from Angular 4 onwards animation is no more a part of the @angular/core library, but is a separate package that needs to be imported in app.module.ts.

- To start with, we need to import the library with the below line of code

```
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
```

- The **BrowserAnimationsModule** needs to be added to the import array in **app.module.ts** as shown below –

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppRoutingModule , RoutingComponent} from './app-routing.module';
import { AppComponent } from './app.component';
import { NewCmpComponent } from './new-cmp/new-cmp.component';
import { ChangeTextDirective } from './change-text.directive';
import { SqrtPipe } from './app.sqrt';
import { MyserviceService } from './myservice.service';
import { HttpClientModule } from '@angular/common/http';
import { ScrollDispatchModule } from '@angular/cdk/scrolling';
import { DragDropModule } from '@angular/cdk/drag-drop';
import { ReactiveFormsModule } from '@angular/forms';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
```

```
@NgModule({
   declarations: [
      SqrtPipe,
      AppComponent,
      NewCmpComponent,
      ChangeTextDirective,
      RoutingComponent
   ],
   imports: [
      BrowserModule,
      AppRoutingModule,
      HttpClientModule,
      ScrollDispatchModule,
      DragDropModule,
      ReactiveFormsModule,
      BrowserAnimationsModule
   ],
   providers: [MyserviceService],
   bootstrap: [AppComponent]
})
export class AppModule { }
```

- In **app.component.html**, we have added the html elements, which are to be animated.

```
<div>
   <button (click) = "animate()">Click Me</button>
   <div [@myanimation] = "state" class = "rotate">
      <img src = "assets/images/img.png" width = "100" height = "100">
   </div>
</div>
```

- Let us now see the **app.component.ts** where the animation is defined.

```
import { Component } from '@angular/core';
import { FormGroup, FormControl, Validators} from '@angular/forms';
import { trigger, state, style, transition, animate } from '@angular/animations';

@Component({
   selector: 'app-root',
   templateUrl: './app.component.html',
   styleUrls: ['./app.component.css'],
   styles:[`
      div {
         margin: 0 auto;
         text-align: center;
         width:200px;
      }
      .rotate {
         width:100px;
         height:100px;
         border:solid 1px red;
      }
   `],
   animations: [
      trigger('myanimation',[
         state('smaller',style({
            transform : 'translateY(100px)'
         })),
         state('larger',style({
            transform : 'translateY(0px)'
         })),
         transition('smaller <=> larger',animate('300ms ease-in'))
      ])
   ]
})
export class AppComponent {
   state: string = "smaller";
   animate() {
      this.state= this.state == 'larger' ? 'smaller' : 'larger';
   }
}
```

- We have to import the animation function that is to be used in the .ts file as shown above.

```
import { trigger, state, style, transition, animate } from '@angular/animations';
```

Here we have imported trigger, state, style, transition, and animate from @angular/animations.

Now, we will add the animations property to the @Component () decorator –

```
animations: [
   trigger('myanimation',[
      state('smaller',style({
         transform : 'translateY(100px)' })),
      state('larger',style({
         transform : 'translateY(0px)' })),
         transition('smaller <=> larger',animate('300ms ease-in'))
   ])
```

- Let us now see the .html file to see how the transition function works −

```
<div>
   <button (click) = "animate()">Click Me</button>
   <div [@myanimation] = "state" class = "rotate">
      <img src = "assets/images/img.png" width = "100" height = "100">
   </div>
</div>
```

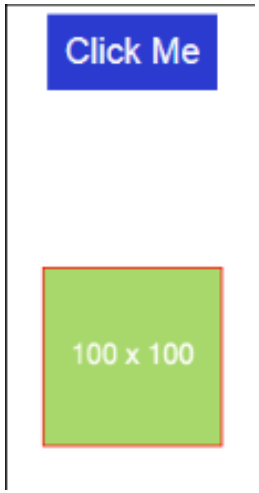
- There is a style property added in the @component directive, which centrally aligns the div. Let us consider the following example to understand the same −

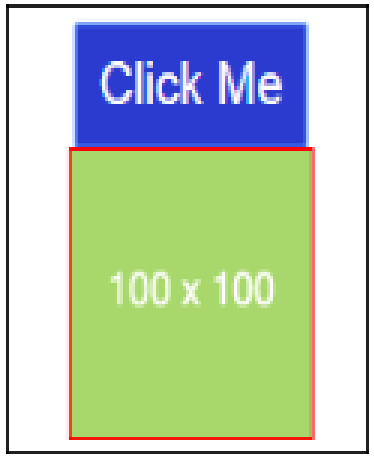
```
styles:[`
   div{
      margin: 0 auto;
      text-align: center;
      width:200px;
   }
   .rotate{
      width:100px;
      height:100px;
      border:solid 1px red;
   }
`],
```

- Here, a special character [``] is used to add styles to the html element, if any. For the div, we have given the animation name defined in the **app.component.ts** file.
- On the click of a button it calls the animate function, which is defined in the **app.component.ts** file as follows –

```
export class AppComponent {
   state: string = "smaller";
   animate() {
      this.state = this.state == 'larger'? 'smaller' : 'larger';
   }
}
```

- This is how the output in the browser **(http://localhost: 4200/)** will look like –





Materials

- Materials offer a lot of built-in modules for your project. Features such as autocomplete, datepicker, slider, menus, grids, and toolbar are available for use with materials in Angular 7.
- To use materials, we need to import the package. Angular 2 also has all the above features but they are available as part of the **@angular/core module**. From Angular 4, Materials module has been made available with a separate module @angular/materials. This helps the user to import only the required materials in their project.
- Following is the command to add materials to your project −

```
npm install --save @angular/material
```

- We will now import the modules in the parent module - **app.module.ts** as shown below.

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppRoutingModule , RoutingComponent} from './app-routing.module';
import { AppComponent } from './app.component';
import { NewCmpComponent } from './new-cmp/new-cmp.component';
import { ChangeTextDirective } from './change-text.directive';
import { SqrtPipe } from './app.sqrt';
import { MyserviceService } from './myservice.service';
import { HttpClientModule } from '@angular/common/http';
import { ScrollDispatchModule } from '@angular/cdk/scrolling';
import { DragDropModule } from '@angular/cdk/drag-drop';
import { ReactiveFormsModule } from '@angular/forms';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { MatButtonModule, MatMenuModule, MatSidenavModule } from '@angular/material';

@NgModule({
   declarations: [
      SqrtPipe,
      AppComponent,
      NewCmpComponent,
      ChangeTextDirective,
      RoutingComponent
   ],
   imports: [
      BrowserModule,
      AppRoutingModule,
      HttpClientModule,
      ScrollDispatchModule,
      DragDropModule,
      ReactiveFormsModule,
      BrowserAnimationsModule,
      MatButtonModule,
      MatMenuModule,
      MatSidenavModule
   ],
   providers: [MyserviceService],
   bootstrap: [AppComponent]
})
export class AppModule { }
```

- In the above file, we have imported the following modules from **@angular/materials**.

```
import { MatButtonModule, MatMenuModule, MatSidenavModule } from '@angular/material';
```

- And the same is used in the imports array as shown below

```
imports: [
   BrowserModule,
   AppRoutingModule,
   HttpClientModule,
   ScrollDispatchModule,
   DragDropModule,
   ReactiveFormsModule,
   BrowserAnimationsModule,
   MatButtonModule,
   MatMenuModule,
   MatSidenavModule
],
```

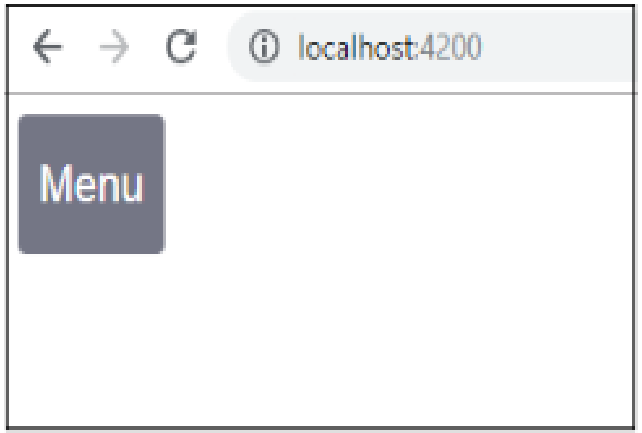
- The app.component.ts is as shown below –

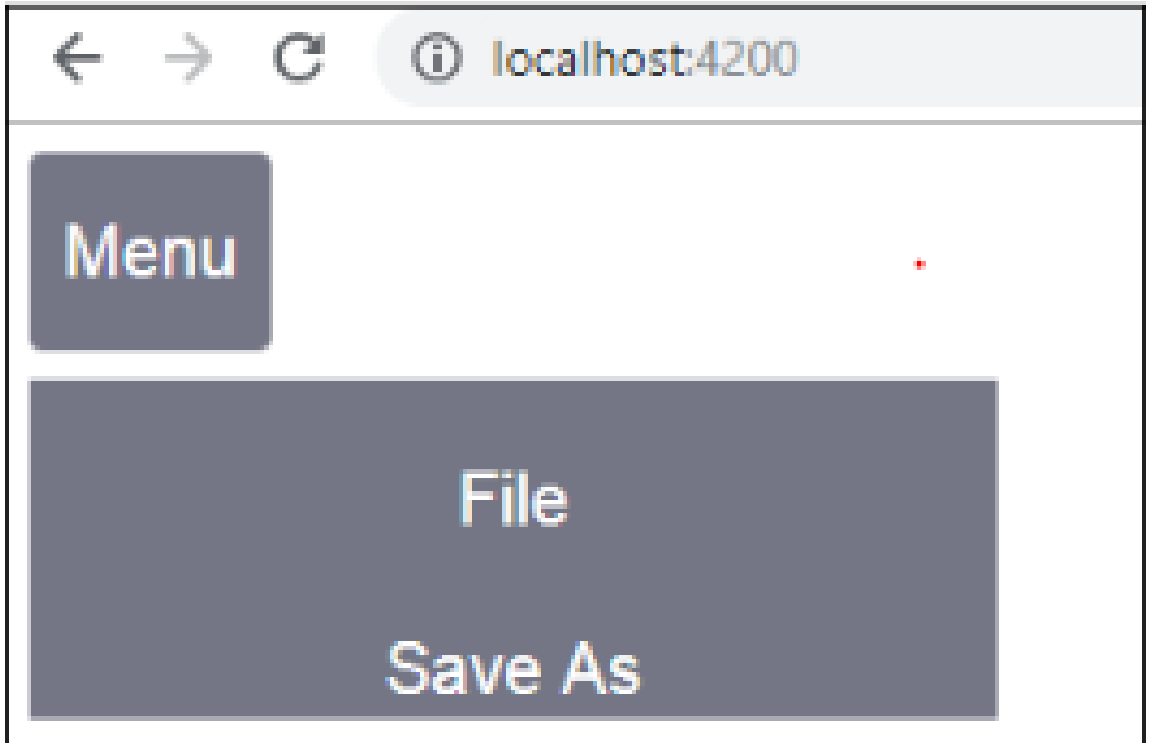
```
import { Component } from '@angular/core';
@Component({
   selector: 'app-root',
   templateUrl: './app.component.html',
   styleUrls: ['./app.component.css']
})
export class AppComponent {
   constructor() {}
}
```

- Let us now add the material-css support in **styles.css**.

```
@import "~@angular/material/prebuilt-themes/indigo-pink.css";
```

- To add menu, **<mat-menu></mat-menu>** is used. The **file** and **Save As** items are added to the button under mat-menu. There is a main button added **Menu**. The reference of the same is given the **<mat-menu>** by using **[matMenuTriggerFor]="menu"** and using the menu with **# in<mat-menu>**.
- The below image is displayed in the browser –




localhost:4200
Menu
File
Save As

Angular Guards

- There are 4 different types of guards.
- ->CanActivate
- ->CanActivateChild
- ->CanDeactivate
- ->Resolve
- ->CanLoad