Developing an App using Angular 4

by N C Patro
Saturday 28th October 2017

Outline

- What is Angular and Why Angular
- Angular1 vs Angular2 vs Angular4
- Environment Setup
- Angular Module, Component, Decorator
- Data Binding and Pipes
- Angular Service and Dependency Injection
- Communicate with Rest APIs
- Angular Routing

What is Angular ?

Angular is a JavaScript framework for client side application development.

Angular is for both mobile and desktop application.

HTML, CSS and JavaScript knowledge is enough to get started with Angular.

Angular is open source

Why Angular

Extends the HTML elements and attributes so its expressive.

Powerful Data Binding for data integrating with view.

Modular design helps to create reusable blocks.

Built in services for most used features like back-end integration.

It provides libraries for unit test and end2end test as well like protractor.

Why Angular 4

Fast (fast app load, faster change detection, faster rendering time)

Modern (adopted latest JS features, supports latest and legacy browser)

Easy (less concepts, less in-built directives, simplified APIs, easy bindings)

Incredible Tooling (help and feedback in IDE, CLI to create app, CLI to generate code blocks, Lint integration)

Language Choice

JavaScript - ES5 and ES6(ES2015)

ES2015 has to be transpiled to ES5, so that all browsers can understand it.

TypeScript

Its superset of JavaScript, it also has to be transpiled to JS.

Dart

Non Javascript language.

TypeScript

Superset of JavaScript

Can be transpiled or compiled to JavaScript

Open Source

Strong Typing

ES2015 features are available like Class, inheritance

IDE support with all feature like Autocomplete and Intellisense

Available as npm package

Editor (IDE) Support

Visual Studio Code

Atom

WebStorm

Eclipse

Bracket

Environment Setup

Install Node and NPM

Node Package Manager

It helps to install packages, libraries and application through command line

Verify that you are running at least node 6.9.x and npm 3.x.x by running node -v and npm -v in CL

Install Angular CLI through npm (npm install -g @angular/cli)

Create a new Project using angular CLI (ng new my-app)

Go to the project directory and start the server (ng serve)

Create Angular App

Create Angular App using CLI

Understand the generated folder structure and files

Build and Run on Browser

Angular Module

Angular Module helps to organize the application.

Every Angular app will have one angular module called app module.

Angular app can have multiple modules based on feature.

Can have shared or common module as well which can be shared by other modules.

Each module have few components and related services associated with it.

Angular Component

Angular application is made of few components and some services which provides functionalities across those components.

Component is comprised of template which is view and class which is associated with view.

Template is created with HTML and angular directives (directive helps in data binding)

Class consists of properties which represents data for view and methods which performs actions for view.

Component also have metadata which holds additional information about component for angular

Angular Decorator

Class becomes angular component after providing the metadata and decorator helps to achieve that.

Decorator is a function that adds metadata to a class and its members.

Always prefixed with @

Angular provides built in decorators like @component()

Component Example

```
import {Component} from '@angular/core';

@Component({
    selector: 'my-app',
    template: `
    <div>
        <h1>{{pageTitle}}</h1>
        <div>My First Angular Component</div>
    </div>`
})
export class AppComponent {
    pageTitle: string = 'Welcome to meetup';
}
```

Angular Module Example

```
import {NgModule} from '@angular/core';
import {BrowserModule} from '@angular/platform-browser';

import {AppComponent} from './app.component';

@NgModule({
    imports: [BrowserModule],
    declarations: [AppComponent],
    bootstrap: [AppComponent]
})
export class AppModule {}
```

Component Template

Template is created with HTML and angular directives (directive helps in data binding)

Component can have Inline Template

Or Linked Template with property templateUrl

Component can have custom directives in template

Directive are custom HTML element or attribute to extend our application HTML vocabulary.

Angular also having built-in directives.

Data Binding

Interpolation (it is a one way binding, from component class to html element)

{{userName}}

{{20*4+2}}

{{‘Name:’ userName }}

{{‘Name: ’ getUserName()}}

Details: {{userId}} - {{userName}}

<input type={{inputType}} />

Structural Built-in Directive *ngIf and *ngFor

```
<table class='table' *ngIf='meetups && meetups.length'>
    <thead>⋯
    </thead>
    <tbody>
        <tr *ngFor='let meetup of meetups'>
            <td>{{ meetup.meetupName }}</td>
            <td>{{ meetup.description }}</td>
            <td>{{ meetup.meetupDate }}</td>
            <td>{{ meetup.price }}</td>
            <td>{{ meetup.starRating }}</td>
        </tr>
    </tbody>
</table>
```

Property and Event Binding

```
<td>
  <img *ngIf='showImage'
  [src]='meetup.imageUrl'
  [title]='meetup.meetupName'
  [style.width.px]='imageWidth'
  [style.margin.px]='imageMargin'>
</td>

<button class='btn btn-primary' (click)='toggleImage()'>
    {{showImage ? 'Hide' : 'Show'}} Image
</button>
```

Two way Binding

```
 <div class='col-md-4'>
   <input type='text' class="form-control" [(ngModel)]='listFilter' />
 </div>

<div class='col-md-6'>
  <h3>Filtered by: {{listFilter}} </h3>
</div>
```

Angular pipes

Pipes transform bind data before displaying in view

Angular has few in-built pipes like date, currency, number etc

Angular allows to write custom pipes as well

```
<td>{{ meetup.meetupName | uppercase }}</td>
<td>{{ meetup.description }}</td>
<td>{{ meetup.meetupDate }}</td>
<td>{{ meetup.price | currency:'INR':true }}</td>
<td>{{ meetup.starRating }}</td>
```

Angular Service and Dependency Injection

Service is a class which will have certain business logic.

Services are used for common logic or features.

It provides common data and logic can be shared across components.

Angular services are singleton means one instance across app.

Angular in-built Injector helps to create and manage singleton services.

Dependency Injection is a pattern in which a class gets the instance of another dependent class.

Angular service example

```
import {Injectable} from '@angular/core';

@Injectable()
export class MeetupListService{

    getMeetupList() {
        let meetups = […
        ];
        return meetups;
    }
}
```

Service Provider

Provider helps to register the service and returns service class.

To register the service it has to be defined as part of angular component or angular module metadata.

```
@NgModule({
  declarations: [...
  ],
  imports: [...
  ],
  providers: [MeetupListService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Angular service injection

```
import { Component } from '@angular/core';
import { MeetupListService } from './meetup-list.service';

@Component({
    selector: 'my-meetups',
    templateUrl: './meetup-list.component.html'
})
export class MeetupListComponent {
    constructor(private _meetupListService: MeetupListService){
    }
}
```

Best Practice of Component

Define Interface or Model

Component specific styles

- Component can have inline styles or styleUrls

LifeCycle Hooks

- Component has a lifecycle managed by Angular.
- Angular creates it, renders it, creates and renders its children, checks it when its data-bound properties change, and destroys it before removing it from the DOM.
- ngOnInit, ngOnChanges, ngOnDestory, ngDoCheck

Retrieve data from REST APIs

Observables and RxJS (Reactive Extensions)

- Observables helps to achieve asynchronous pattern
- Angular event system uses observables and RxJs library
- In ReactiveX an observer subscribes to an Observable, then that observer reacts to whatever item or sequence of items the Observable emits.
- Observables are different from promises.
- Observables supports multiple methods like map, filter, reduce.

Angular routing

Single page application

Each Component will have route configuration

Routes gets activated based on user action

Once the url changes, angular looks for routing configuration and shows the component view which matches the url.

Thank You