HOW

Hands On Workshop

By : Vijay Shivakumar

What do we need ?

Hardware and software

IDE : webstorm / atom / visual studio code
Browsers : chrome latest
Platform : nodejs latest
Database : mongodb / firebase
Version Control : git
Network : internet access to download
from git and npmjs.org

Objective

About this course

Understand and explore ES6 / ES7

Write Programs using Typescript 2.7

Understand members of Angular bundle

Develop programs using Angular platform

Workflow for fast Angular application creation with Angular CLI

Unit Testing Angular code

About | me

Vijay Shivakumar

Designer | Developer | Trainer

Adobe | CERTIFIED EXPERT Flex® with AIR

Training & Consultation of
Contemporary Web Technologies and Adobe products from past 14 years

About | you

Designer

Developer

Architect

Business Analyst

Technology Enthusiast

What's Angular

What is it ?

Designer

Developer

Architect

Business Analyst

Technology Enthusiast

Angular | Features

Leverages on new HTML5 Features
Includes cutting edge JavaScript features ES6, ES7
TypeScript for strong data typing
Better error handling
Speed and performance
Modular approach
Hybrid (Mobile, Tablet and Web support)
Feature rich to create SPAs
(DOM handling, 2 way Binding, Routing, Animation, Validation, Ajax, RESTful API)

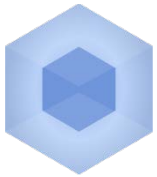
Setup / Tooling

What is required...

NodeJS
TypeScript
TraceurJS
BableJS
SystemJS
Webpack
Angular/cli
Express
MongoDB Mlab ID / Firebase ID
Karma

Git
NodeMon
NVM

Build Tools | Using Angular-CLI



ES5	None
ES6	Traceur BableJS SystemJS Webpack
TypeScript	Traceur BableJS SystemJS Webpack

Architecture

What make Angular ?

One way data flow : Data flow only from parent to child unlike angular1

Dependency Injection : Resources are passed to a component as and when required

Components : Creates a custom tag where the component replaces its content

Directives : Adds new functionality to HTML elements that they never had

Templates : Are separate from component

Zone.js : Manages change detection

Rendering Targets : Render an app for many devices with

Browser-platform

Browser-platform-dymanic

Module

What is it doing..?

Is different from ES6 module

Every application must have at least 1 module (root module)

Root module is decorated with 'NgModule' from @angular/core

import : [FirstModule, SecondModule],

declarations : [Components, Pipes, Services, Directives]

providers: [servicesToInject1, servicesToInject2]

bootstrap : [mainComponent]

Component | What is it ?

A basic Component has two parts.

1. A Component decorator
2. A component definition class

Component Decorator :

We can think of decorators as metadata added to our code.
When we use @Component on a class,
we are “decorating” that class as a Component.

Component Class :

Will have a constructor, properties , methods and life cycle events by default

Component

What is it doing..?

A component is a combination of a view (the template) and some logic

Is decorated with 'Component' from @angular/core

By convention every application must have a main component which is bootstrapped via module.

selector: 'aDomElement'(usually a custom tag name)

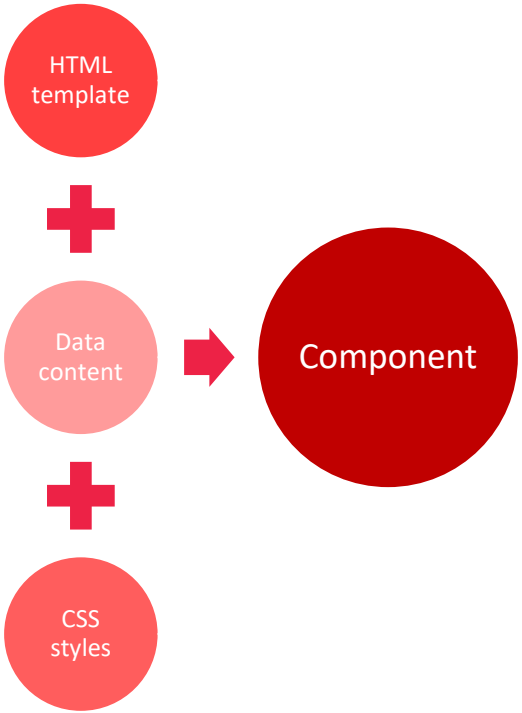
template:

templateUrl:

styles:[]

stylesUrl:[]

Component | What is it ?



Templates | View

template : Inline

templateUrl : external

Display Data

Format Data

User Interaction

Avoid business logic

Styling | Component v/s Page

Styling a component

styles:[`

selector { property : value; property : value }

selector { property : value; property : value }

`]

OR

stylesUrl:[“location/style.css”, “anotherlocation/style.css”]

Styling a page

That you define in the html page in the head or body section

Binding Interpolation

Property Binding

```
<h1>{{ 2+2 }}</h1>
<h1>{{ user.name }}</h1>
<h1 [textContent]="user.name"></h1>
<h1 bind-innerHTML='user.name'></h1>
<h1>{{  user.message()  }}</h1>
```

Keep it simple and fast (they should not take more time to compute)

Avoid multiple statements

You can not use assignment operators e.g. = , +=, ++, - - in property binding

You can not create an object e.g. new User().

Binding Interpolation

Property Binding

```
<h1>{{ 2+2 }}</h1>
<h1>{{ user.name }}</h1>
<h1 [textContent]="user.name"></h1>
<h1 bind-innerHTML='user.name'></h1>
<h1>{{  user.message()  }}</h1>
```

Tip: Use ? to handle undefined
Use the safe navigation operator
e.g. <h1>{{ no-user?.prop }}</h1>

Keep it simple and fast (they should not take more time to compute)

Avoid multiple statements

You can not use assignment operators e.g. = , +=, ++, -- in property binding

You can not create an object e.g. new User().

Event Binding

Statement Binding

```
<button (click)="callfun()"> Click Me </button>
```

Keep it simple and fast (they should not take more time to compute)

The callback functions can take a single parameter which is referred as $event

(If you wish to send multiple params you can wrap them in an object and send)

Avoid business logic on templates

You can not create an object e.g. new User().

Events

Element events supported

mouseenter	dragover	cut
mousedown	drop	copy
mouseup	focus	paste
click	blur	keydown
dblclick	submit	keypress
drag	scroll	keyup

Style Binding | class / ngClass

[class.className] =“stronghero”

[ngClass]={ expression that returns a string, object or an array}

In the example below both stronghero and boxclass can be applied if it matches the conditions

Eg;

[ngClass]="{ stronghero: heroPower > 5, boxclass: rating > 0.5}

Tip: conditionally applied classes will append to existing classes

Style Binding | style/

[style.color] =“#333”

[ngStyle]={ expression that returns a style value}

In the example below the ternary operator will return a style property and value combination

Eg;

[ngStyle]="{‘color’: heroPower > 5 ? ‘green’ : ‘red’}”

Tip: conditionally applied classes will append to existing classes

Structural Directive

Repeat / If / hidden

Adds or removes the DOM contents or they change the structure of DOM
hence the name

```
<ul>
    <li *ngFor = “let user of users”> </li>
</ul>
```

In this example it shall loop over and for users

and create a temp variable user in the scope of the loop

For every loop the li and its content is repeated

<li *ngIf=“true/false”>{{ data }} OR <li [hidden]=“true/false”>{{ data }}

Structural Directive

Repeat / If / hidden

Adds or removes the DOM contents or they change the structure of DOM
hence the name

```
<ul>
    <li *ngFor = “let user of users”> </li>
</ul>
```

In this example it shall loop over and for users

and create a temp variable user in the scope of the loop

For every loop the li and its content is repeated

<li *ngIf=“true/false”>{{ data }} OR <li [hidden]=“true/false”>{{ data }}

Tip: Usage of hidden is efficient

Structural Directive | Switch

```
<div [ngSwitch]="hero.power">
        <h1 *ngSwitchCase="8">Strong</h1>
        <h1 *ngSwitchCase="7">Weak</h1>
        <h1 *ngSwitchDefault>Recovering</h1>
</div>
```

Pipes

Format your output

AsyncPipe	will deal with observable values and display latest result
DatePipe	used to format the date as needed
LowerCasePipe	converts to lowercase
UpperCasePipe	converts to uppercase
CurrencyPipe	applies a currency symbol and manage integer and decimals
DecimalPipe	manages decimal values

Communication | Input & Outputs

@Input decorator from @angular/core

allows data inlet in to the component

@Output decorator from @angular/core

allows data to be sent from the component

Tip : only events are allowed to be used as an output

The same can be done with these properties of a component

input :

output :

You can use template variables to communicate from a child component to parent component

Services

Dealing with Data

Simple classes that fetch some data

@Injectable decorator required if you inject other built-in service

Attach the service in the module as a provider so that every component can have access to it..

Inject it in to any component you may want to use the service

Routing

Navigating / Multipage

Angular supports HTML5 and Hash based URL Routing

Define Base Path

Import Router

Configure Routes

Place Templates

Activate Routes

Thank you

vijay.shivu@gmail.com