**Angular 2 Notes**

TO start the application -- npm start

ctrl c to stop

ng g c componentname -- command to create a new component using cli

ng g s servicename – command to create a service

ng g d directive name – create custom directive

interpolation -- string interpolation{{string}}

dependency injection

injecting or providing the dependencies of class into its constructor.

Property binding: [property name] = property value; ex <img [src] =<http://img.png> /> -- One way binding(component to view)

Attribute binding [attr.property name]

Event bubbling – event bubbles up in a dom tree.

Event binding -- (keyup.enter) = “methodname($event)” or you can write an expression

Template variables: #tempVariable will reference the element: used to simplify code

ngModel – Two way binding ex [(ngModel)] = propertyName -- ngModel imported from forms module

pipes : currency pipe, date pipe, decimal pipe used to format data.

Custom pipes

@Input and @Output Properties

@Input() isSelected: boolean;

Input properties to pass input or state to a component

Output properties to raise events from these components. Following are the steps to raise a custom event from component:

1. Add (change) event to html and in the component add the event that u want to be called once the change is done
2. Declare output property in the component where u have declared input output properties and initialize it with eventemitter.

@Output() change= new EventEmitter();

1. In Onclick event raise an event this.change.emit()

View Encapsulation : shadow dom

Ng-content – to replace with custom content in html. Provide custom content to reusable controller

Ng-container – render something without putting it under div

**Directives**

\* marks the structural directive – to modify structure of DOM

\* tells that angular converts the statement into ng-template and manually adds conditions.

\*ngIf – removes the element from dom if cond not satisfied

From ang4 <ng-template> introduced so that you can write ngIf condition based.

<ng-template #coursesList>

\*ngIf = ”condition>0; then coursesList else course”

[hidden] – hides the element in dom should be used in case of small dom tree or if building the tree is costly affair

[ngSwitch] = bind value

\*ngSwitchCase – structural directive -- changes structure of dom

for of iterates over iterable

for in iterates over properties of object

\*ngFor=”let course of courses; trackBy:trackCourse”

trackBy – is used when you don’t want the dom to rerender every time on button click. It will track by whatever id returned from trackCourse method.

Safe Traversal object : task.assignee?.name – if assignee is not null then name otherwise null

**Custom Directives**

HostListner: listen to the events raised from the dom elements like onBlur or onFocus

Elementref add in constructor – to access the property of dom element

Pass data to a directive using input properties

Selector of the directive can also be used to pass data by making it as an attribute[] and pass data from it

ngOnchanges

passing data to a nested component thru @input parameter

u can declare a property of nested components (kind of directives) as [property name]

-------------------------------

Template driven forms – create directives, good for simple forms, easy to create

Reactive forms – create separate controls, good for complex forms and unit testable

Template driven: ngModel need to declare name attrib f or input elements

Validations: firstName.errors.required

FormControl : ngModel and formGroup : ngModelGroup

ngForm used for submit buton

[ngValue] can be used to bind complex object

Reactive forms

Abstract control is the base class for FormControl and FormControlGroup

formControlName and formGroup are directives used to bind the reactive form

use validators.required in component to do validation as a property of FormControl

--------------------------------

lifecyclehooks

http://thisdavej.com/how-to-watch-for-files-changes-in-node-js/

subscribing to observable

to upgrade angular library versions:

<https://update.angular.io/>

https://github.com/dotnetcurry