**Prerequisites**

* Angular language service: vs code extension
* Augury: Browser extension for debugging
* Angular CLI

**Concepts in Angular**

* COMPONENTS: User defined tag in HTML
* DIRECTIVES: User defined attribute/property
* PIPES: Data transformation or data formatting
* SERVICES:
  + API calls
  + Business logic
  + Data sharing or state management
* Each component has 4 files:
  + ts: Controller. Variables declared inside ts files are called as Model Variables
  + HTML: View
  + CSS: Component specific styles
  + spec.ts: unit test
* Commands:
  + ng g component <name>
  + directive
  + pipe
  + service
* Two types of components
  + SMART: Container (containers folder). Does logical part.
  + DUMP: presentation (component folder). Does only presentation part.
  + A component can talk to its parent and child only.
  + Structure of component
    - SMART
      * DUMP
  + PARENT TO CHILD: @Input property
  + CHILD TO PARENT: @Output events
* PIPES syntax: input | expression/function: param1: param2: param3 (additional arguments to pipe)
* @HostListener() to listen to html events
* FORMS: @angular/forms
* TEMPLATE DRIVEN: FormsModule
* REACTIVE / MODEL DRIVEN: ReactiveFormsModule
* FORM STATES:
  + Valid & invalid: state of all the validations
  + Touched & untouched: focusing the field
  + Dirty & pristine: change of the value
  + Errors: identifying which validation has failed
* Model-driven form:
  + Each object is FormGroup
  + Each control is FormControl
  + Validation returns null if it is valid, else object of error
* Lifecycle sequence: <https://angular.io/guide/lifecycle-hooks#lifecycle-sequence>
* SERVICES
  + Promise: JS
  + Observable (recommended): RxJS library
* Limitations of Promise:
  + No debounce: delay time
  + Waiting time
  + No cancel request feature
* Services API calls
  + HttpClientModule: @angular/common/http
  + Ng g s services/product
  + To register service globally declare the service in providers in app.module.ts or write @Injectable decorator in service.ts file with providedIn: ‘root’
* Difference between Promise and Observable:
  + Promise: .then().catch()
  + Observable: .subscribe()
* To use routing in Angular @angular/router: RouterModule is used
* In rxJS, information is exchanged/shared using Subject.
* There are two types of observable: Hot and Cold
  + Hot observable keeps receiving updates.
  + Cold observable receives update just once e.g. API calls.
* Router Guards
* Passing params:
  + RESTful: mandatory: abc.com/productDetail/100
  + Query: optional: abc.com/search?price=1000&brand=value
  + Router for routing
  + activatedRouter for getting params

NOTE: <http://tiny.cc/nmvg7y> for content and references