**D: 02/05/2017**

**Firebase:**

**Firebase** is a backend service that provides data storage, file storage, authentication, and static website hosting for your Angular app

**Webpack:**

**Webpack** is a popular module bundler, a tool for bundling application source code in convenient chunks and for loading that code from a server into a browser.

Webpack can handle the various types of files.

Like .js, .html, .png, .css

**D: 03/05/2017**

**Src**

(Source folder it’s developing the app)

1. **Karma.conf.js** is the **root** of the project (this is for configurations)

Karma is a testing library we can use in Anjular2.

1. **Package.json** is installed libraries for project and package.json is a configuration file which is used by npm(this is nodejs package manager).
2. **Dependencies:**

Means packages require for this project

1. **devDependencies:**

This is for only developing the app not deployment.

1. **Tsconfig.json**:

These files maintain the typescript compilers.

1. Webpack.config.js

**Config**

**(**Entire configuration of the application**)**

1. **Helpers.js** is generates is the webpack
2. **Karma-test-shim.js** this is import all of the libraries need karma
3. **Karma.conf.js** is handles the actually configuration of karma
4. **Webpack.common.js** has commented all configuration js files. We can entry the files from app(polufills,main,vendor)
5. **Webpack.dev.js** is the configuration we are used in developer project
6. **Webpack.prod.js** for generating a production build
7. **Webpack.test.js**  is for testing

**Public**

1. Public/sass
2. **Sass** is the pre compiler for the css (The reason for using the sass in the project give us greater flexibility style of project)
3. Sass/styles.scss **&&** sass/\_settings.scss

**Src**

1. Src/app/**index.html** this is the starting point of the app
2. Src/app/**main.ts** this is for angular2(its run the bootstrap file)
3. Src/app/**polyfills.ts** we import the libraries for app (Extension with **.ts**)
4. Src/app/**vendor.ts**

**Angular2 development languages**

Dart

JavaScript

Typescript

1. App/start/app.component.ts (this is angular2 component)
2. App/start/app.component.css (Component level styles)
3. App/start/app.component.html (Display the messages when app run)
4. App/start/app.component.spec.ts (Karma test file)
5. Src/app/shared/nav.componet.ts
6. Src/app/shared/nav.componet.css
7. Src/app/home/home.componet.ts
8. Src/app/home/home.componet.css
9. Src/app/home/home.componet.html
10. Src/app/error/error.component.ts
11. Src/app/error/error.component.css
12. Src/app/error/error.component.html

App/shared/app.routing.ts

App/admin/admin.module.ts

App/admin/signUp/sign-up.component.ts

App/admin/signUp/sign-up.component.css

App/admin/signUp/sign-up.component.html

1. App/admin/login/login.component.html
2. App/admin/login/login.component.css
3. App/admin/login/login.component.ts
4. App/admin/adminMenu/admin-menu.component.ts (adminMenu for access the different areas of backend)
5. App/admin/adminMenu/admin-menu.component.html
6. App/admin/adminMenu/admin-menu.component.css
7. App/admin/adminComponent/admin.component.ts
8. App/admin/adminShared/user.service.ts

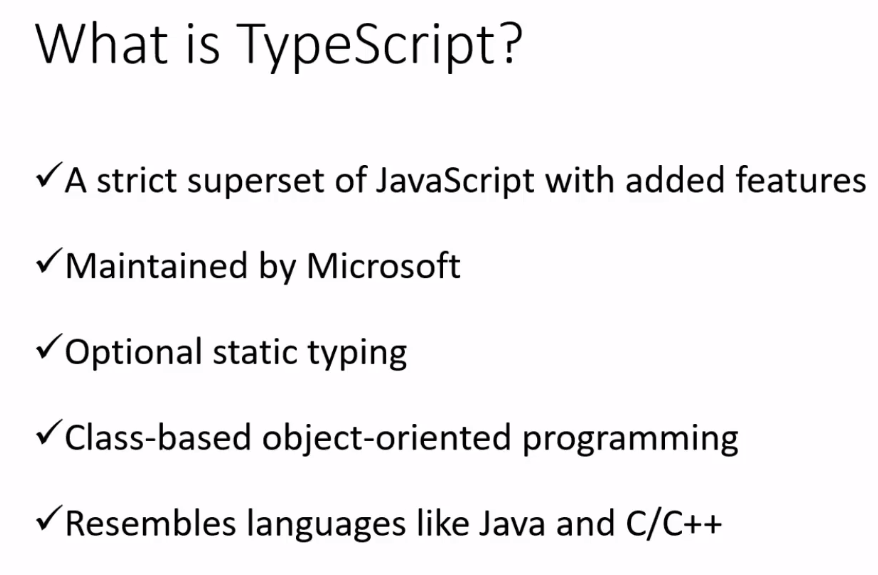
**D: 04/05/2017**

**Angular2 in 60M**

1. What is Angular2?

Angular2 is a js client side framework for creating powerful web applications. The most popular js framework to date.





**Components:**

Main way to build and specify elements and logic on the page.

**Services:**

Services are used for reusable data services to share b/w components throughout an application.

**NPM**: Node Package Modules

**Angular2 Installation in Windows**

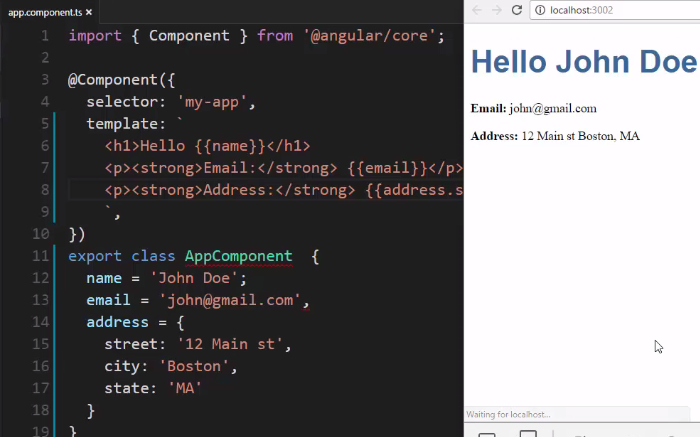
[https://git-scm.com](https://git-scm.com/) (click on Download for Windows)

git clone https://github.com/angular/quickstart my-proj (clone the project form this url)

cd my-proj

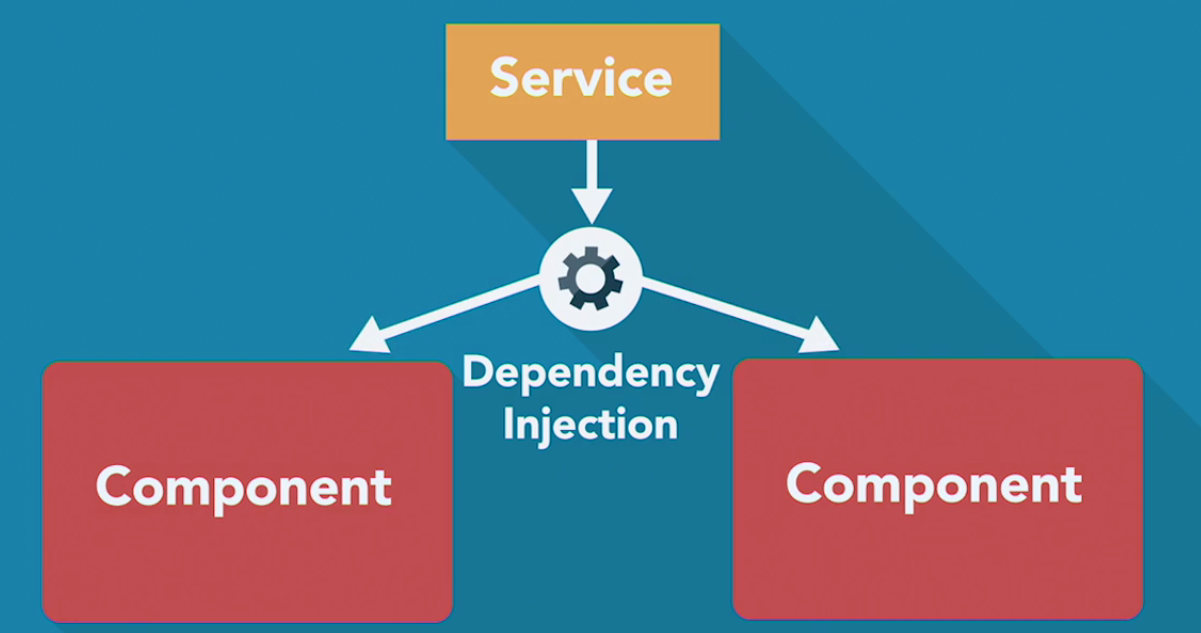
npm install





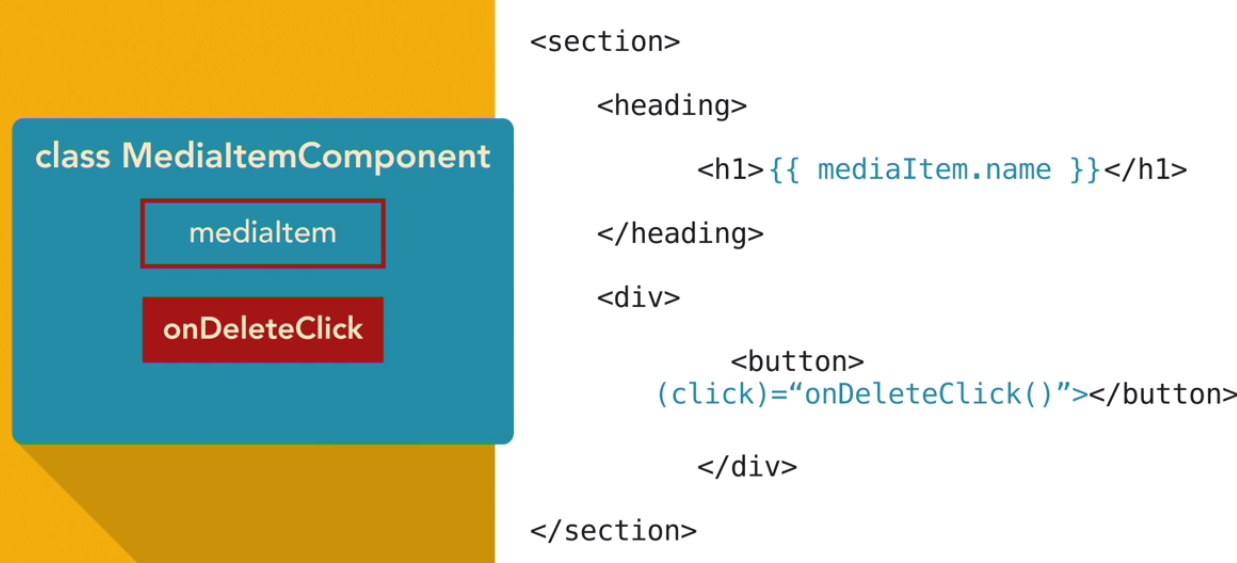
**Angular2 (Lynda)**





Components:

In Angular 2, “everything is a component.” Components are the main way we build and specify elements and logic on the page, through both custom elements and attributes that add functionality to our existing components.



Mediaitem= property name

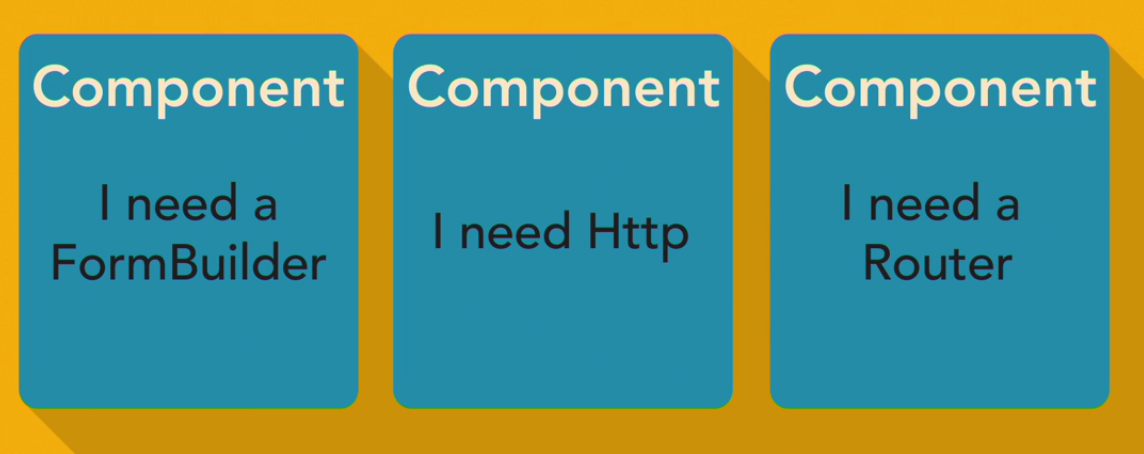
onDeleteClick  method



We can use components with in the components. And nested elements within the elements.

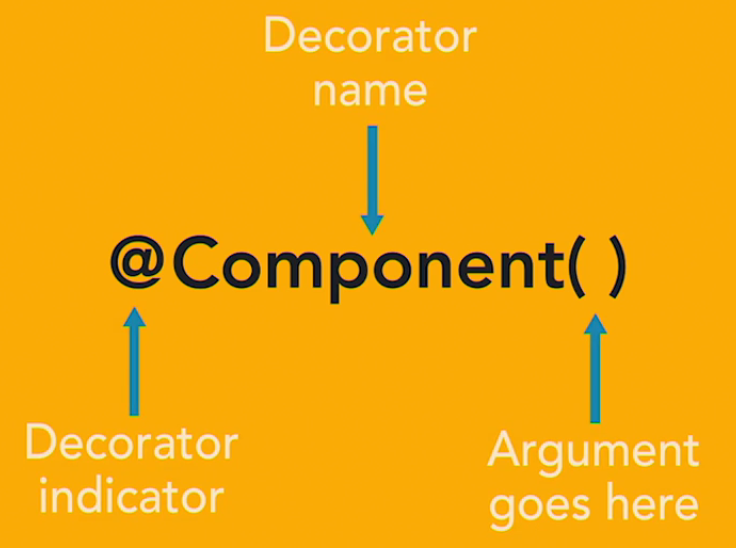
Each component is the configure with selectors. It tells the angular what markup associate component class logic with.





Decorator:

Expression that evaluates to a function allowing annotation of classes at design time.



To decorate a component you need to write two data properties minimum.

* Selector: and template: and templateurl:
* Angular application starts with an angular module.
* Angular module helps to keep application code. Organized by blocks of functionality and features.
* Root module access the starting point of the application.
* **NgModule: NgModules** help organize an application into cohesive blocks of functionality. An NgModule is a class adorned with the @NgModule decorator function. @NgModule takes a metadata object that tells Angular how to compile and run module code. It identifies the module's own components, directives, and pipes, making some of them public so external components can use them. @NgModule may add service providers to the application dependency injectors.

**D: 05/05/2017**

1. App/maint.ts
2. App/app.module.ts
3. App/app.component.ts
4. App/app.component.html

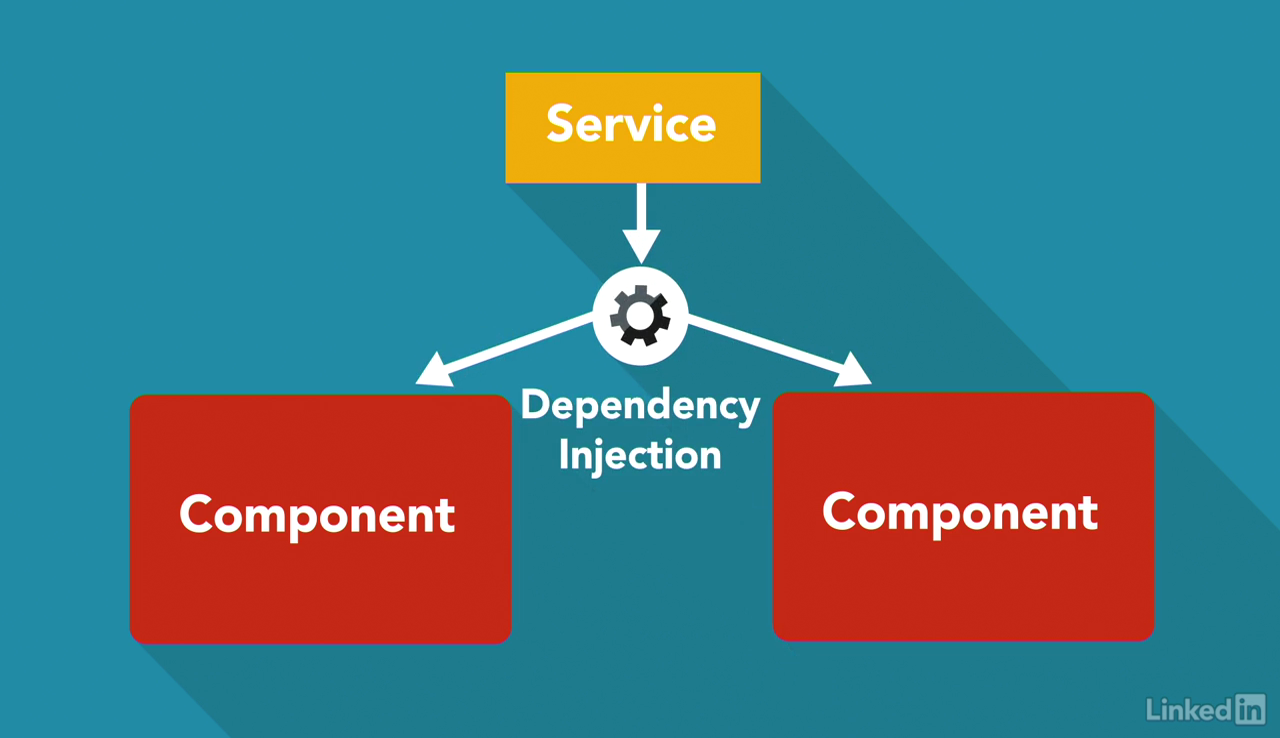
**D: 06/05/2017**

**D:19/05/2017**

1) Angular build a client side application. And angular2 is component architecture.

2) **TypeScript:**

**Type Script** is a language for application-scale JavaScript. TypeScript adds optional types, classes, and modules to JavaScript. TypeScript supports tools for large-scale JavaScript applications for any browser, for any host, on any OS. TypeScript compiles to readable, standards-based JavaScript.







\*\*\* 1 line is Import statement for module's loading.

\*\*\* 3 line is component for typescript decorater.

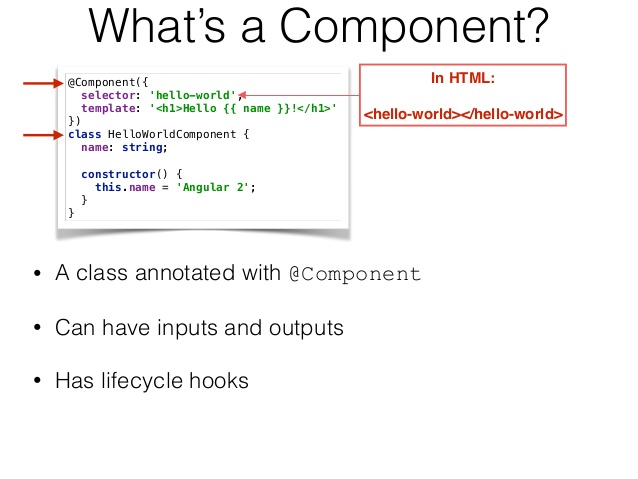
\*\*\* export is a ts key word. And it class. In class we can write constructor. That is ts strong parameter and syntex.

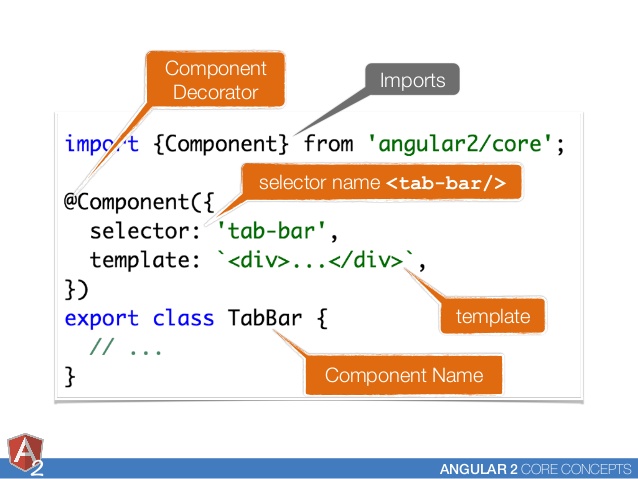
3)

Overall concepts.

**Components:**

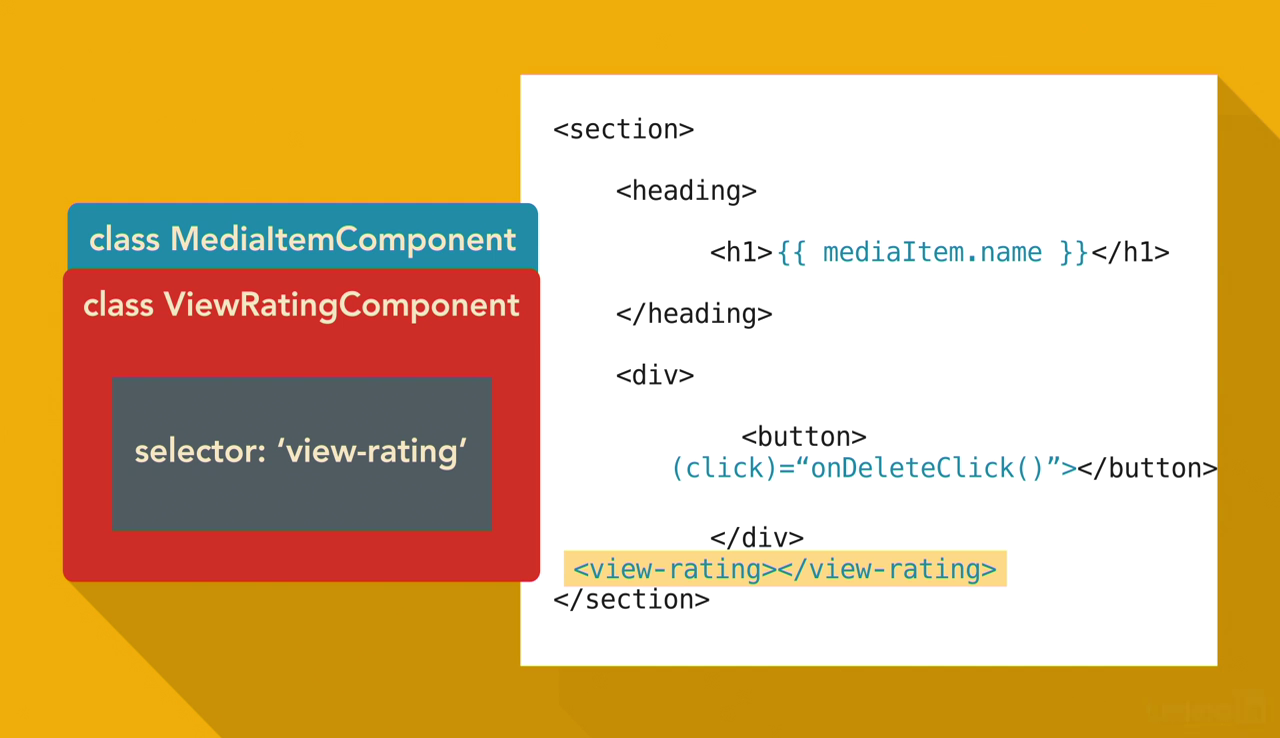
**Component** decorator allows you to mark a class as an **Angular component** and provide additional metadata that determines how the **component** should be processed, instantiated and used at runtime. ... An **Angular** application is a tree of**Angular components**. **Angular components** are a subset of directives.





\*\*\* Component builds the logic for application.

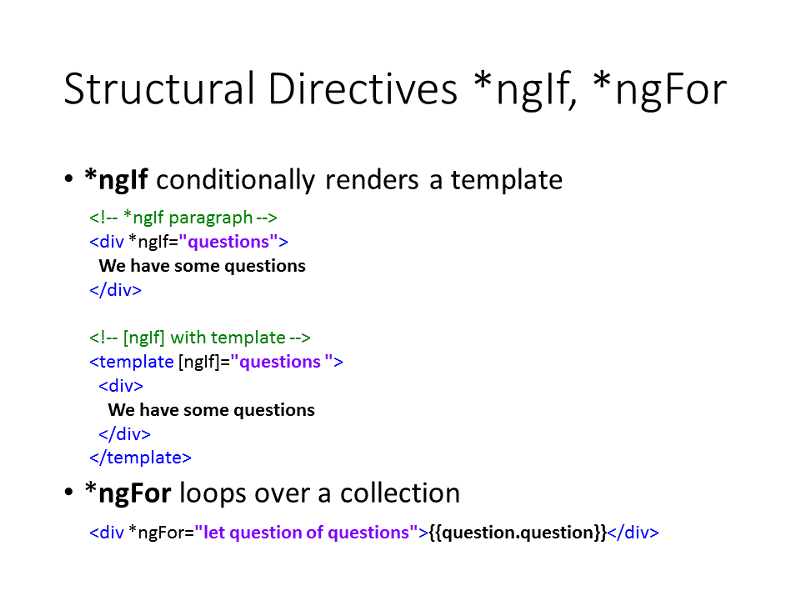
\*\*\* We can use components with in the components.



**Directives: (D**irectives provide the functionality)

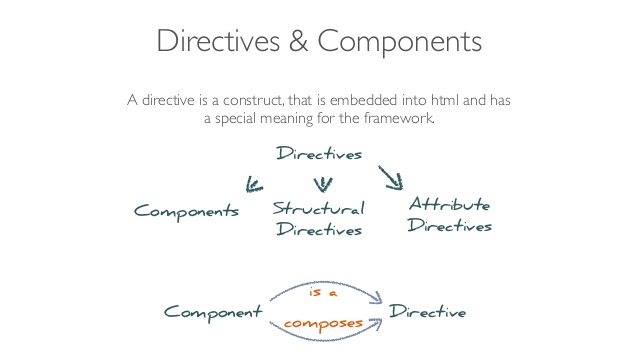
There are three kinds of **directives** in **Angular**: Components—**directives** with a template.

Structural **directives**—change the DOM layout by adding and removing DOM elements.

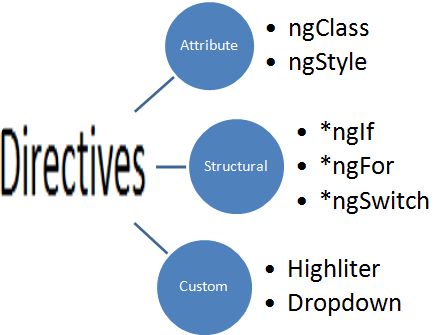


Attribute **directives**—change the appearance or behavior of an element, component, or another directive.



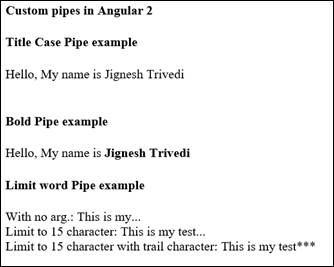


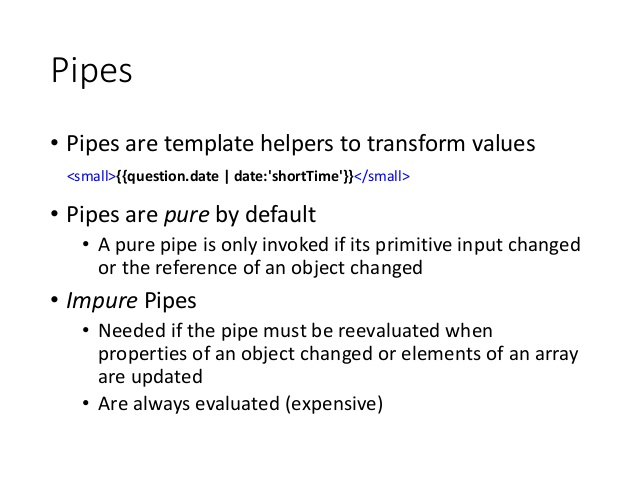
\*\*\* we can use attributes or templates in directives.

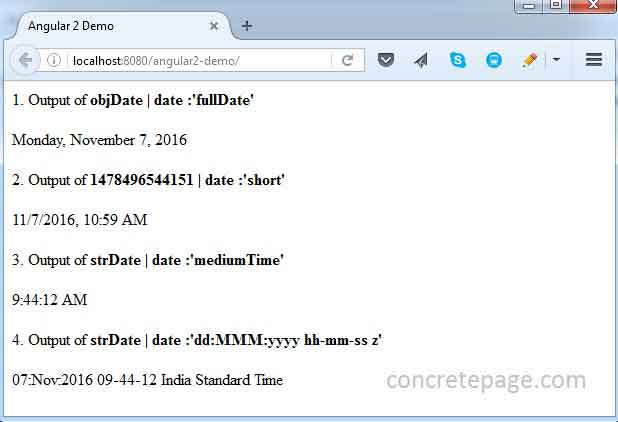


**Pipes:**

**Pipes** transform displayed values within a template. Every application starts out with what seems like a simple task: get data, transform them, and show them to users. Getting data could be as simple as creating a local variable or as complex as streaming data over a WebSocket.

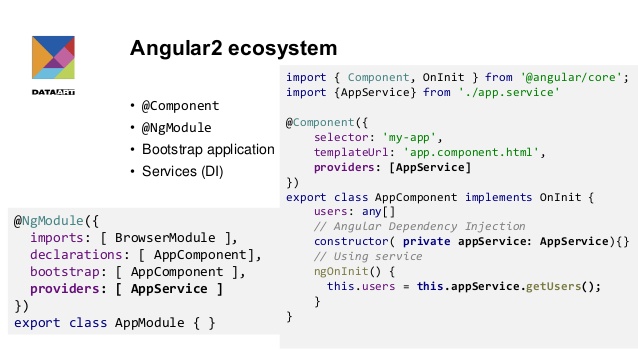






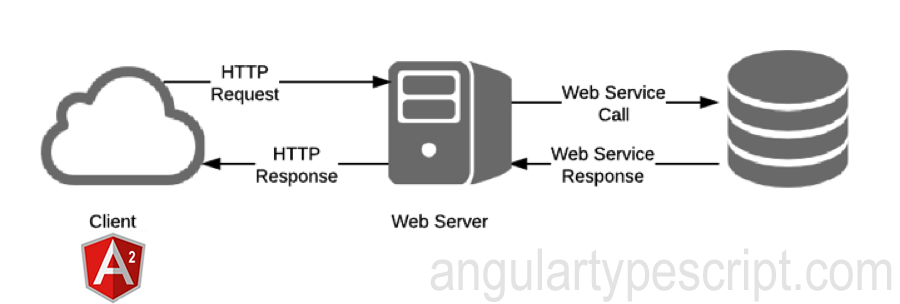
**Services: (**Services passing data to components)

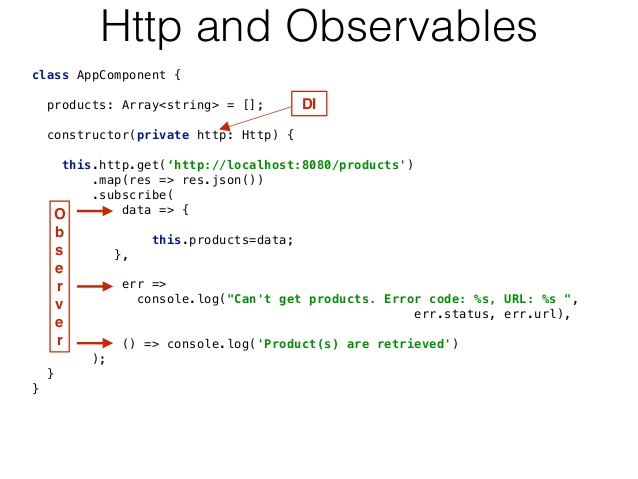
**Services** are JavaScript functions that are responsible for doing a specific task only. **Angular services** are injected using Dependency Injection mechanism and include the value, function or feature which is required by the application.



**HttP:**

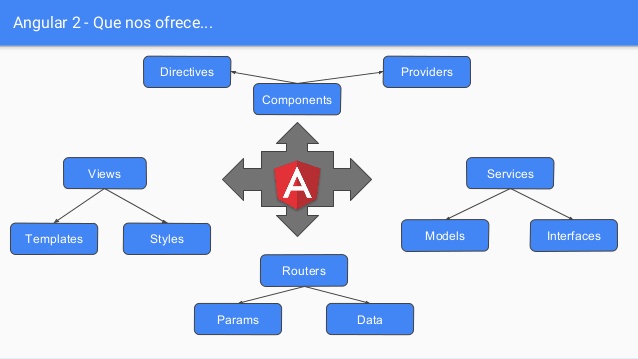
**HTTP** is the primary protocol for browser/server communication. ... Modern browsers support **two HTTP**-based APIs: XMLHttpRequest (XHR) and JSONP. A few browsers also support Fetch. The**Angular HTTP** library simplifies application programming with the XHR and JSONP APIs.

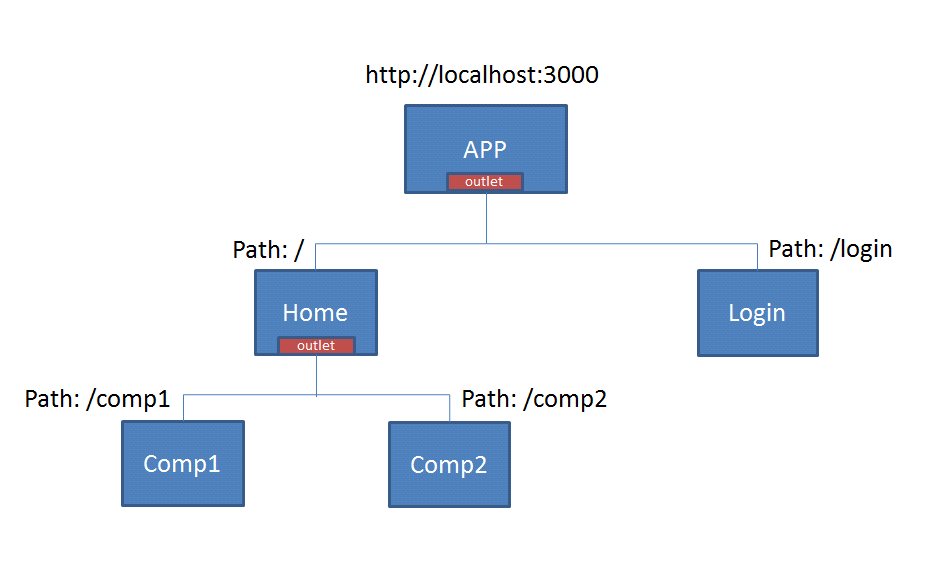




**Routing:**

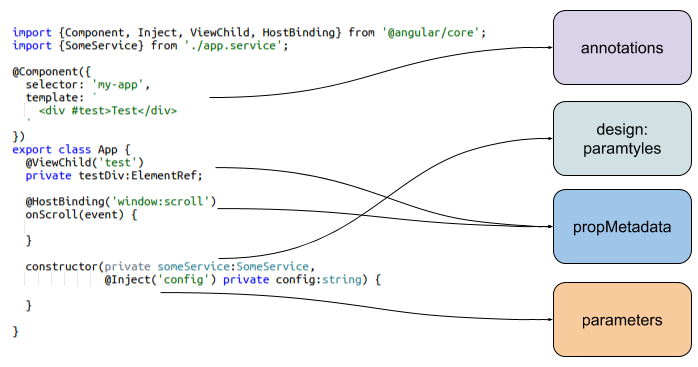
 A component that we route to has access to something that **Angular** calls the ActivatedRoute . An ActivatedRoute is an object that contains information about route parameters, query parameters and URL fragments. ContactsDetailComponent needs exactly that to get the id of a contact.

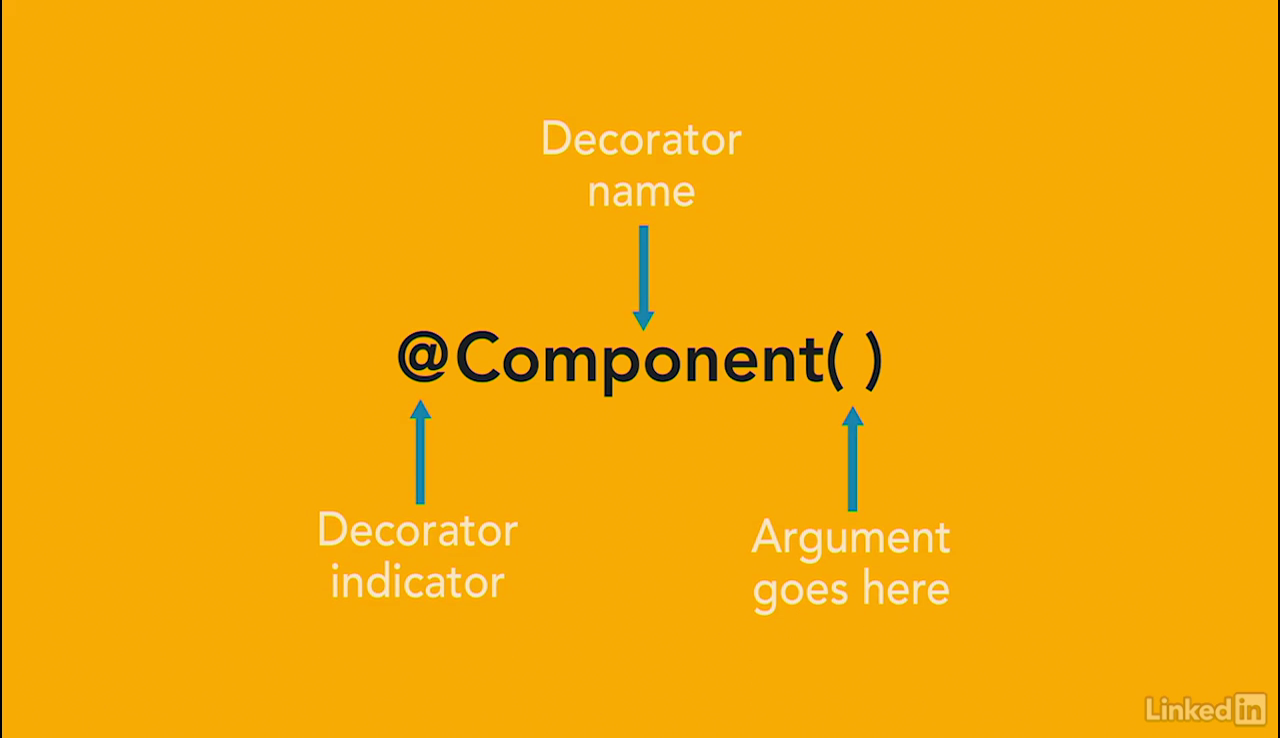


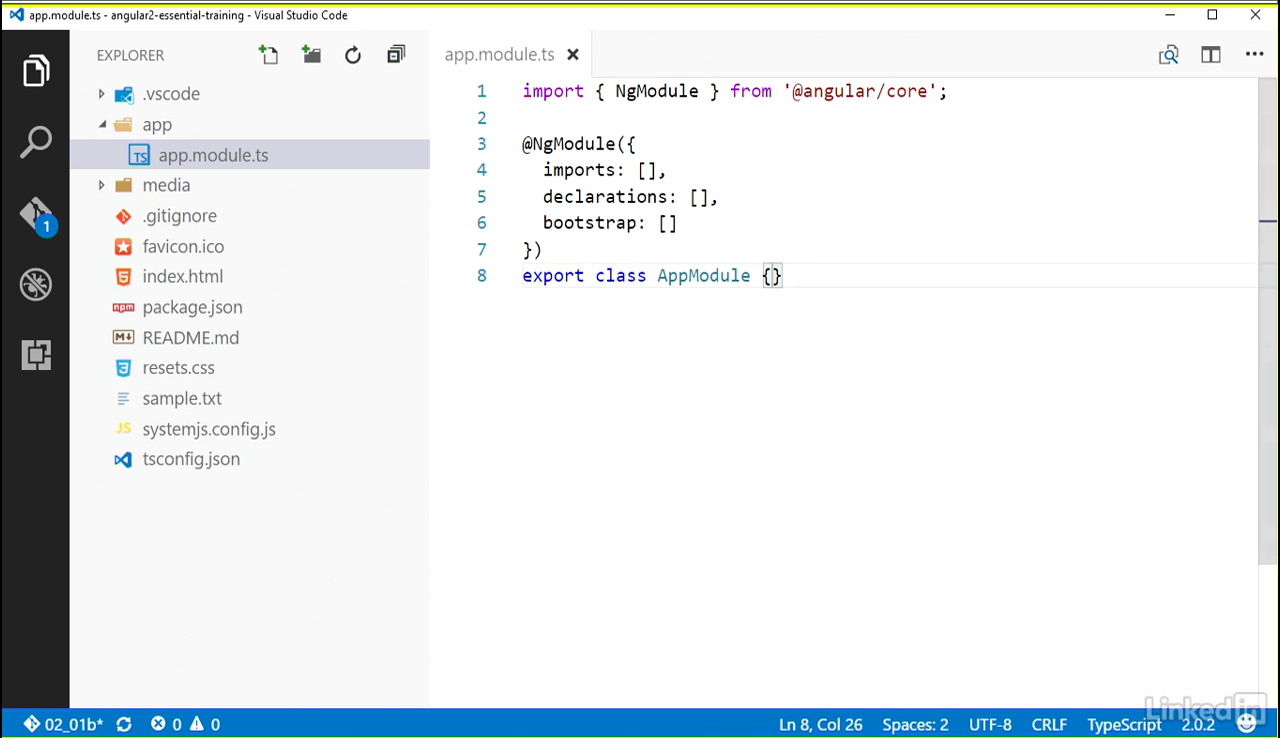
****

**Decorators**.

**Decorators** are proposed for a future version of JavaScript, but the **Angular** team really wanted to use them, and they have been included in Type Script. **Decorators** are functions that are invoked with a prefixed @ symbol, and immediately followed by a class , parameter, method or property.



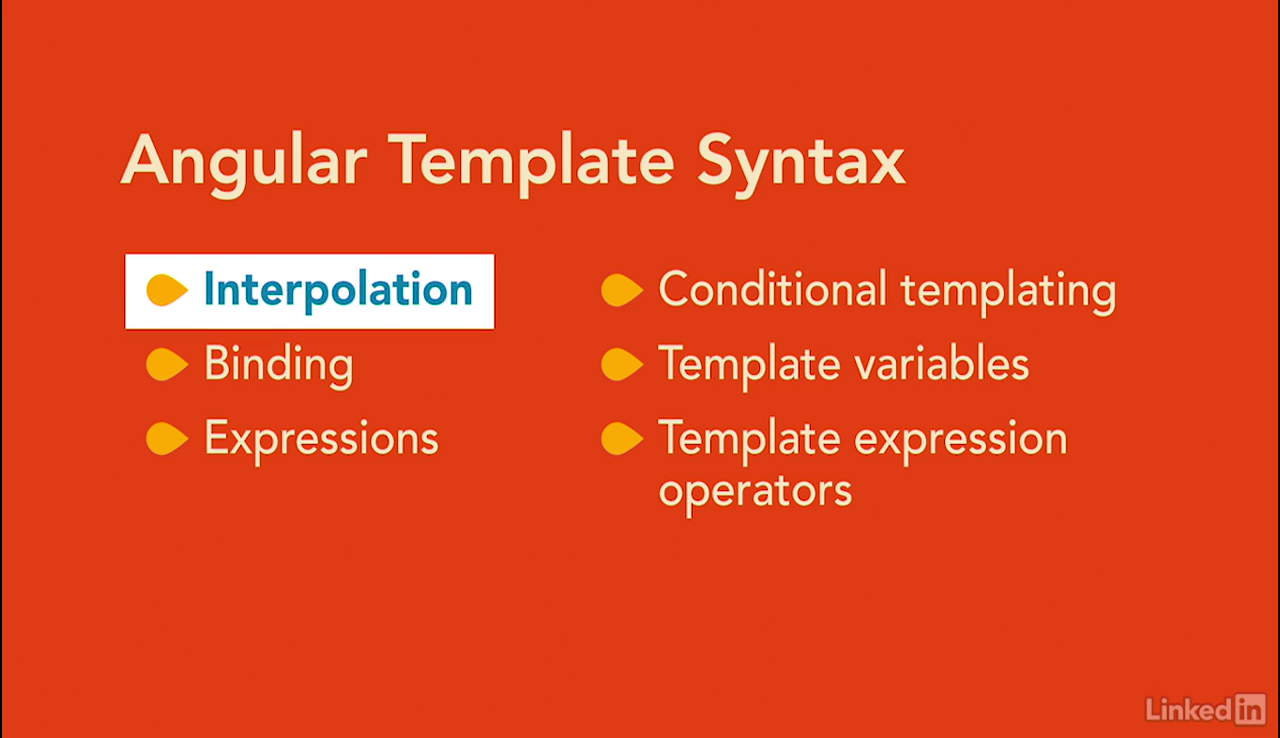
****

****

Angular2 application start from the module's.

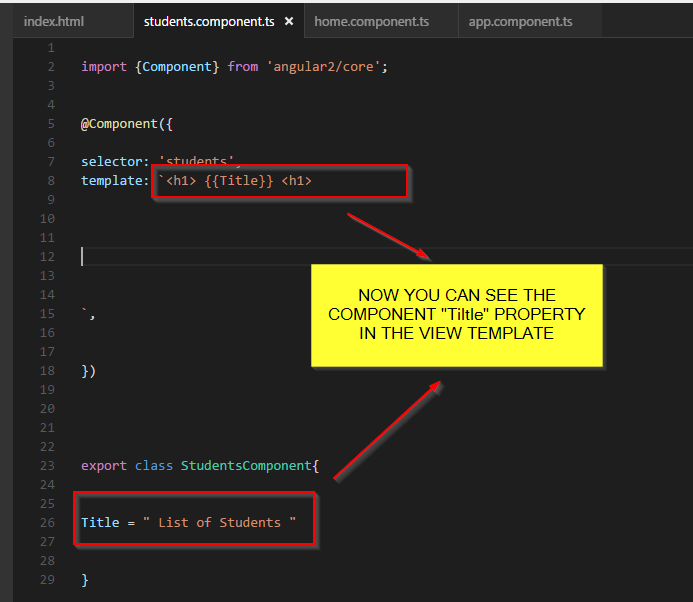
Imports ==> brings the other modules your module need.

Declarations ==>Used to make components and directories, pipes available to your module that don’t come from another module.

Bootstrap ==> Used for root and that angular know which component and components for the starting point of bootstrap classes of your app code.

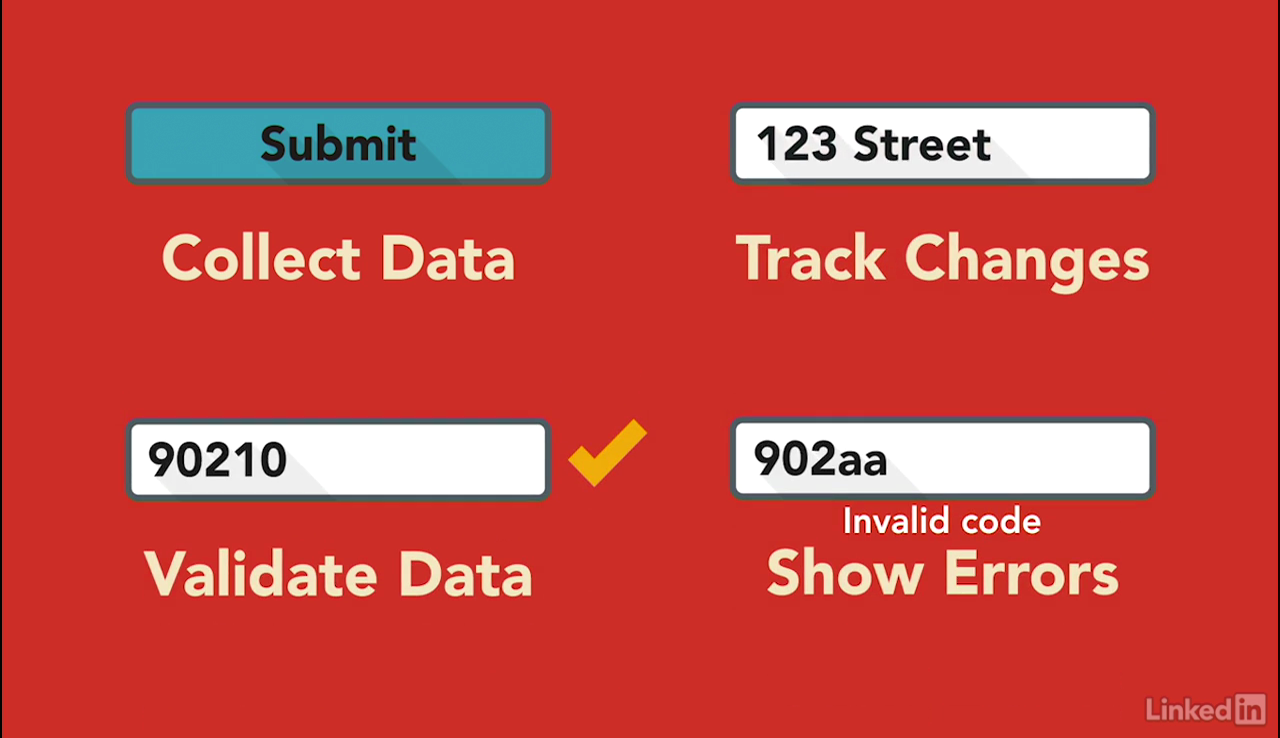
**Interpolation:**

The text between the braces is often the name of a component property. ...**Interpolation** is a special **syntax** that **Angular** converts into a property binding, as is explained below. But first, let's take a closer look at template expressions and statements.

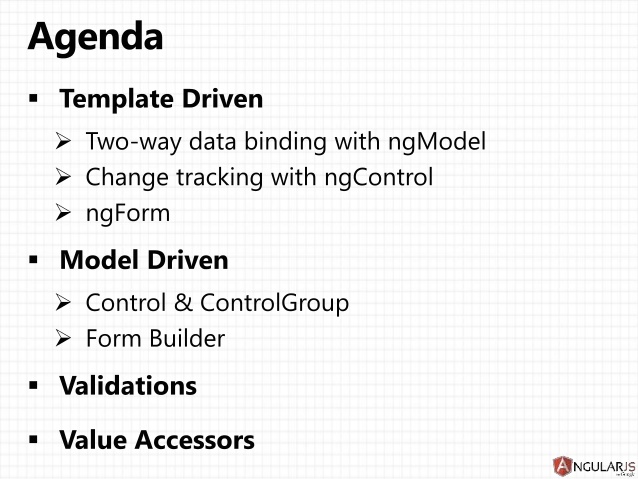


**Forms:**

A **form** creates a cohesive, effective, and compelling data entry experience. An **Angular form**coordinates a set of data-bound user controls, tracks changes, validates input, and presents errors.**Forms** are the mainstay of business applications.



Types of forms.



**Dependency Injection:**

**Angular's dependency injection** system creates and delivers dependent services "just-in-time".**Dependency injection** is an important application design pattern. **Angular** has its own **dependency injection** framework, and you really can't build an **Angular** application without it.

**RouterOutlet** directive is used to display views for a given route. This is where templates of specific routes are loaded while we navigate: