GETTING STARTED WITH ANGULAR

Prakash Dillibabu
(Dteam)

8/10/2017

What is angular ?

Angular is a JavaScript Framework which allows you to create reactive Single-Page-Applications [SPAs]

TypeScript :

Typescript has More features than vanilla Javascript.

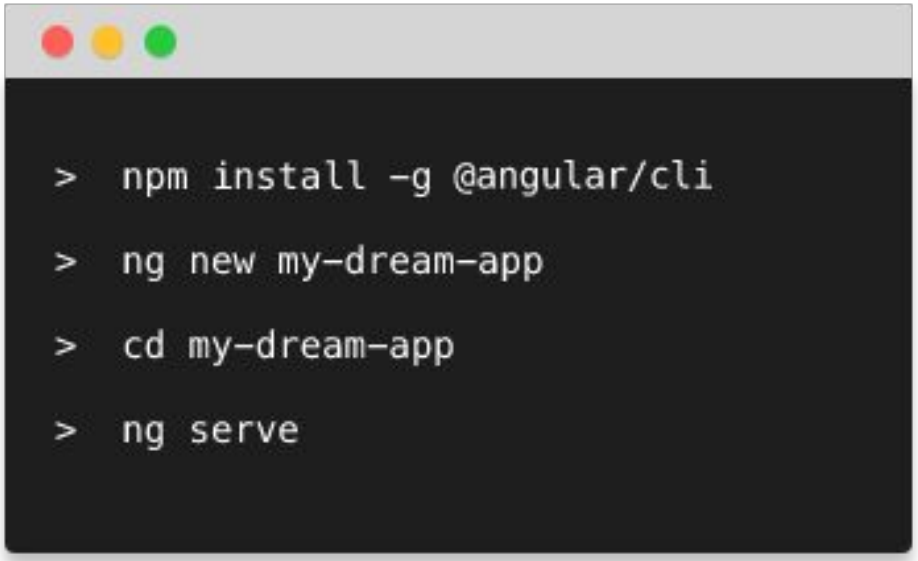
(eg. Types, Classes , Interface)

Angular CLI

Install angular/cli

Create new angular project

Run the project

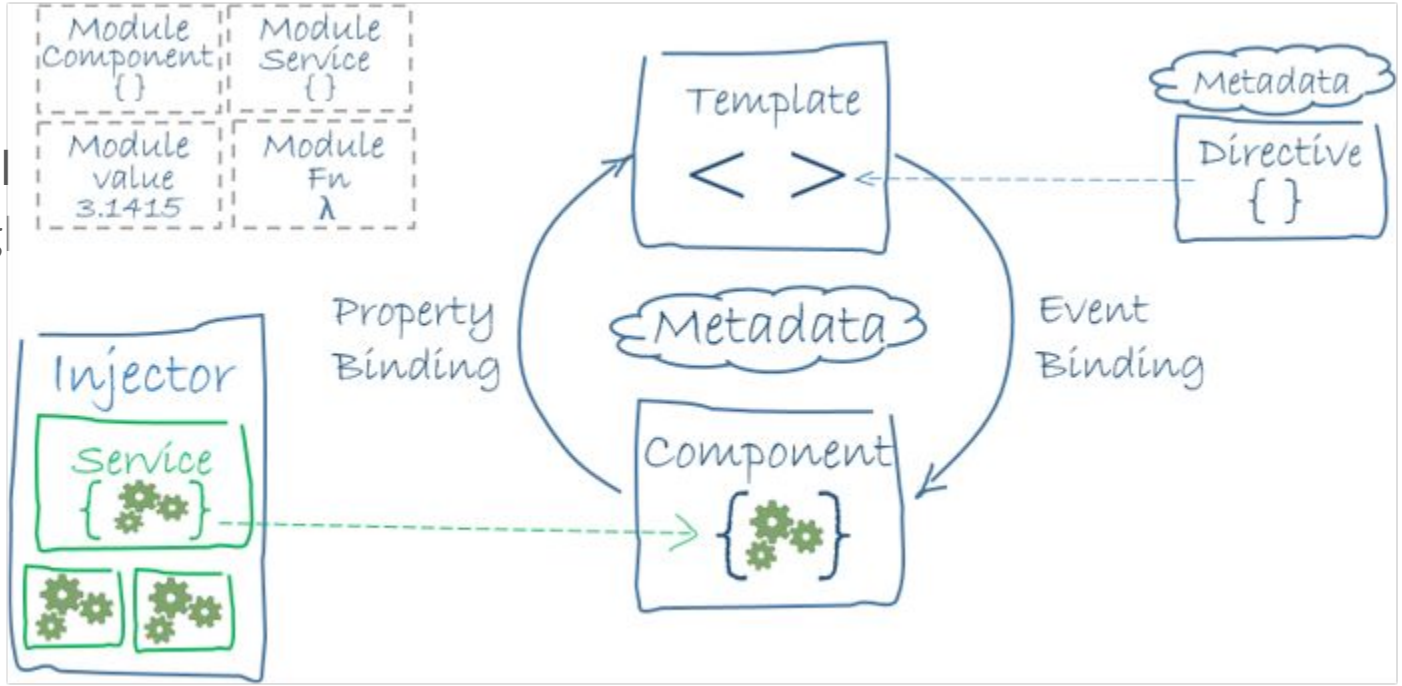


Architecture Overview

- Module
- Components
- Templates
- Metadata
- Data binding
- Directives
- Services
- Dependency injection

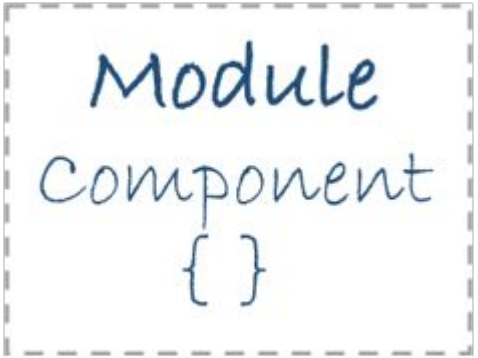
Architecture Overview

Anul
Sing



Module

- Angular apps are modular and Angular has its own modularity system called **NgModules.**
- While the root module may be the only module in a small application, most apps have many more feature modules,
- An NgModule, whether a root or feature, is a class with an @NgModule decorator.



Module

- NgModule is a decorator function that takes a single metadata object whose properties describe the module. The most important properties are:

    1. Declarations
    2. Exports
    3. Imports
    4. Providers
    5. Bootstrap

@Ngmodule ({ })

Decorators are functions that modify JavaScript classes. Angular has many decorators that attach metadata to classes so that it knows what those classes mean and how they should work.

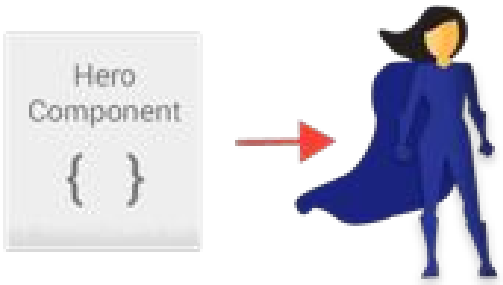
Module

src/app/app.module.ts

```
import { NgModule }      from '@angular/core';
import { BrowserModule } from '@angular/platform-
browser';
@NgModule({
  imports:      [ BrowserModule ],
  providers:    [ Logger ],
  declarations: [ AppComponent ],
  exports:      [ AppComponent ],
  bootstrap:    [ AppComponent ]
})
export class AppModule { }
```

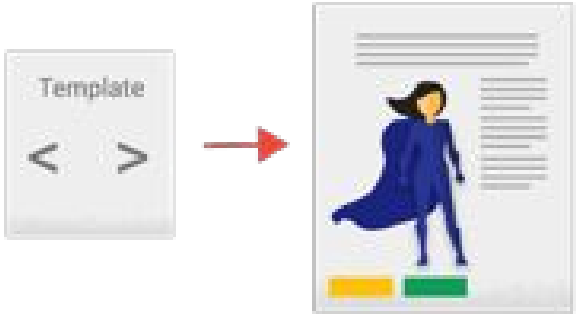
Components

- A component controls a patch of screen called a view.




```
export class HeroListComponent implements OnInit {
  heroes: Hero[];
  selectedHero: Hero;

  constructor(private service: HeroService) { }

  ngOnInit() {
    this.heroes = this.service.getHeroes();
  }

  selectHero(hero: Hero) { this.selectedHero = hero; }
}
```

Templates



- You define a component's view with its companion template. A template is a form of HTML that tells Angular how to render the component.

- A template looks like regular HTML, except for a few differences.

Templates

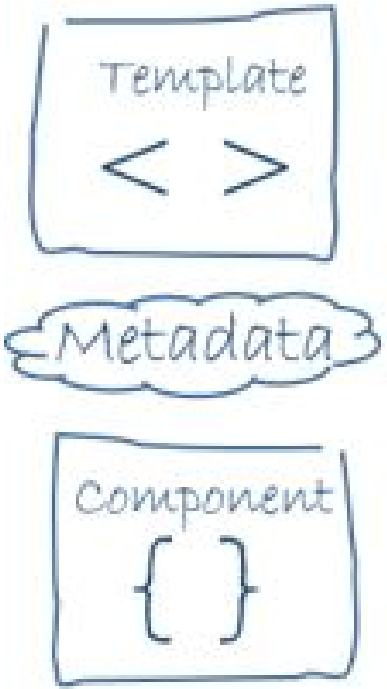
```
<h2>Hero List</h2>

<p><i>Pick a hero from the list</i></p>
<ul>
  <li *ngFor="let hero of heroes"
(click)="selectHero(hero)">
    {{hero.name}}
  </li>
</ul>

<hero-detail *ngIf="selectedHero"
[hero]="selectedHero"></hero-detail>
```

Metadata

- Metadata tells Angular how to process a class.
- Here are a few of the most useful @Component configuration options:
- Selector , templateUrl , providers

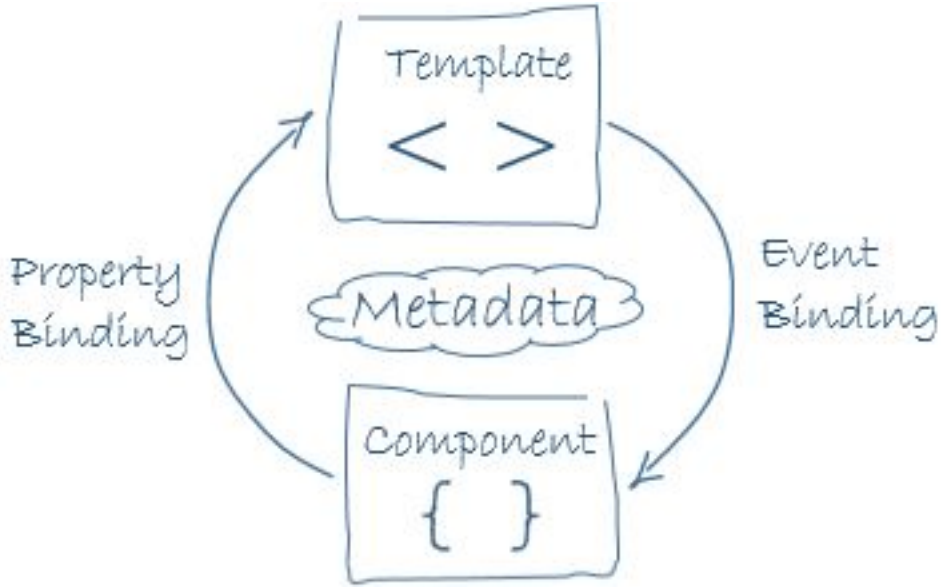
Metadata

- The metadata in the @Component tells Angular where to get the major building blocks you specify for the component.
- The template, metadata, and component together describe a view.
- Apply other metadata decorators in a similar fashion to guide Angular behavior. @Injectable, @Input, and @Output are a few of the more popular decorators.



Data binding

- Angular supports data binding, a mechanism for coordinating parts of a template with parts of a component. Add binding markup to the template HTML to tell Angular how to connect both sides.



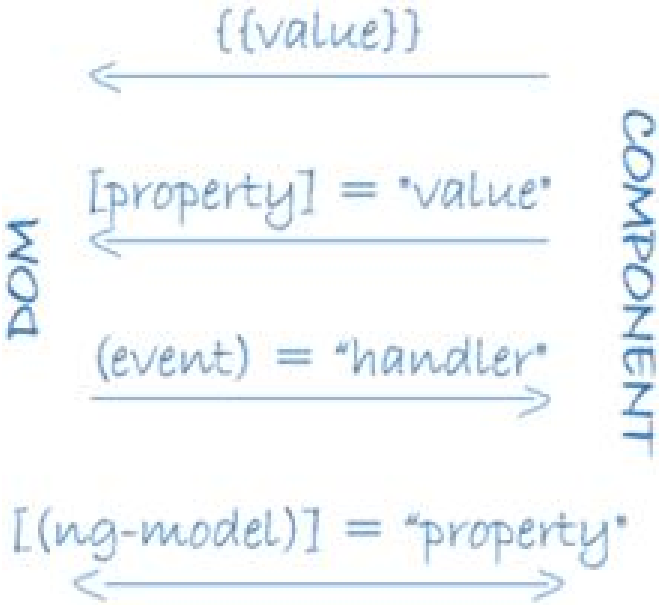
Data binding

String interpolation

Property Binding

Event Binding

Two-way Binding

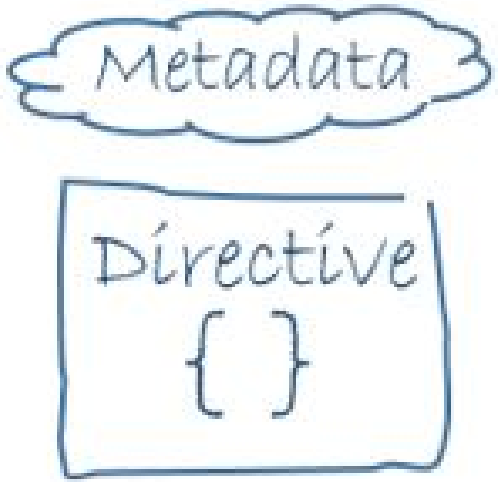


Directives

- Angular templates are dynamic. When Angular renders them, it transforms the DOM according to the instructions given by directives.

```
<li *ngFor="let hero of heroes"></li>
<hero-detail
*ngIf="selectedHero"></hero-detail>
```

*ngFor , *ngIf



Services

- Service is a broad category encompassing any value, function, or feature that your application needs.

- Almost anything can be a service. A service is typically a class with a narrow, well-defined purpose. It should do something specific and do it well.

Dependency injection

- Dependency injection is a way to supply a new instance of a class with the fully-formed dependencies it requires.

- Most dependencies are services. Angular uses dependency injection to provide new components with the services they need.


```
constructor(private service: HeroService) {
}
```