Angular basics

EPAM Systems

Agenda

1. What is Angular?
2. Why use Angular?
3. Angular CLI
4. Angular architecture
5. Modules
6. Components
7. Services and dependency injection

What is Angular and why use it?

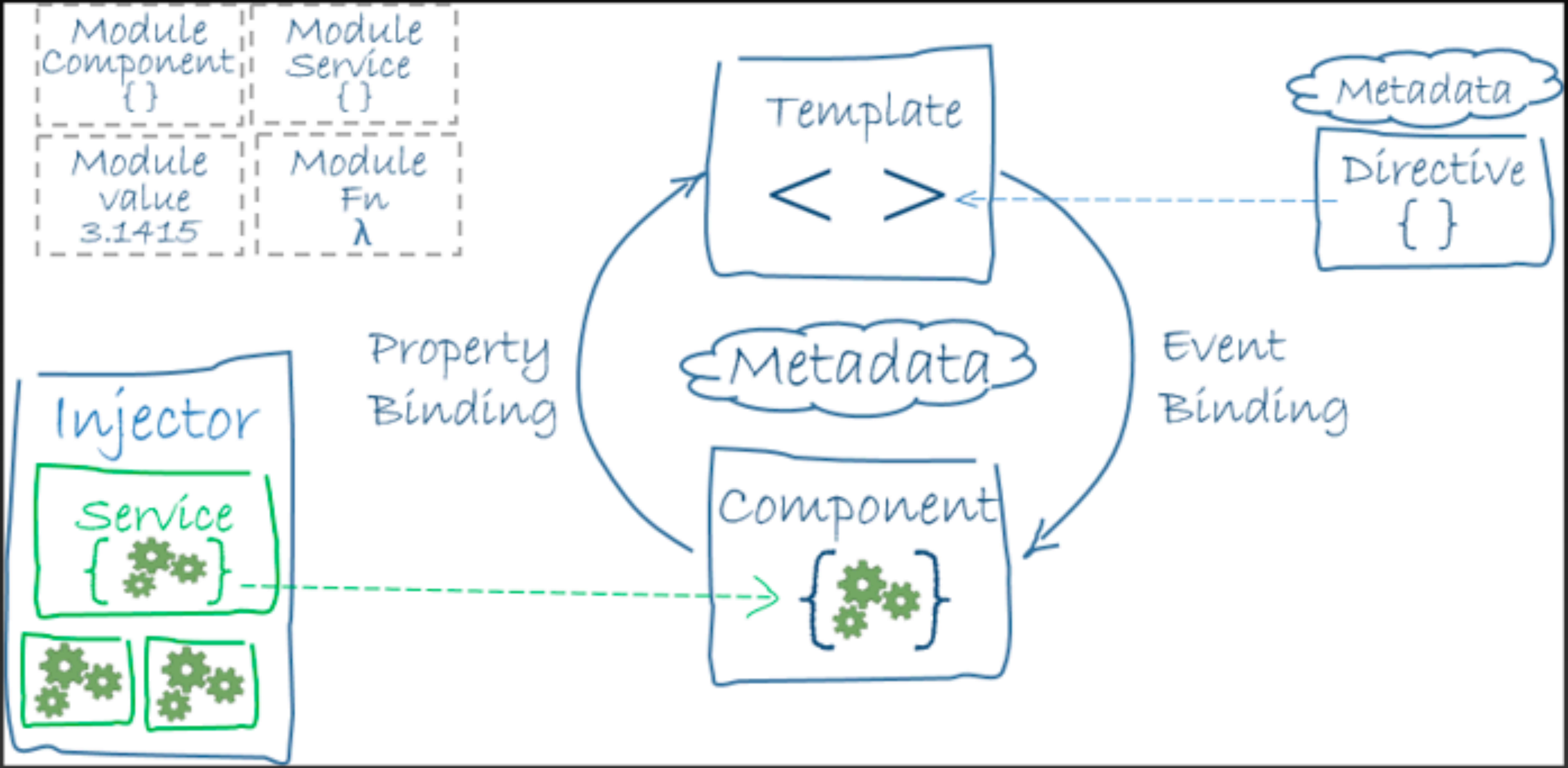
- Angular is a development platform, built on TypeScript. As a platform, Angular includes:
  - A component-based framework for building scalable web applications
  - A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
  - A suite of developer tools to help you develop, build, test, and update your code

Angular CLI

ng new	Creates a new Angular workspace
ng serve	Builds and serves your application, rebuilding on file changes
ng generate	Generates or modifies files based on a schematic
ng build	Compiles an Angular app into an output directory

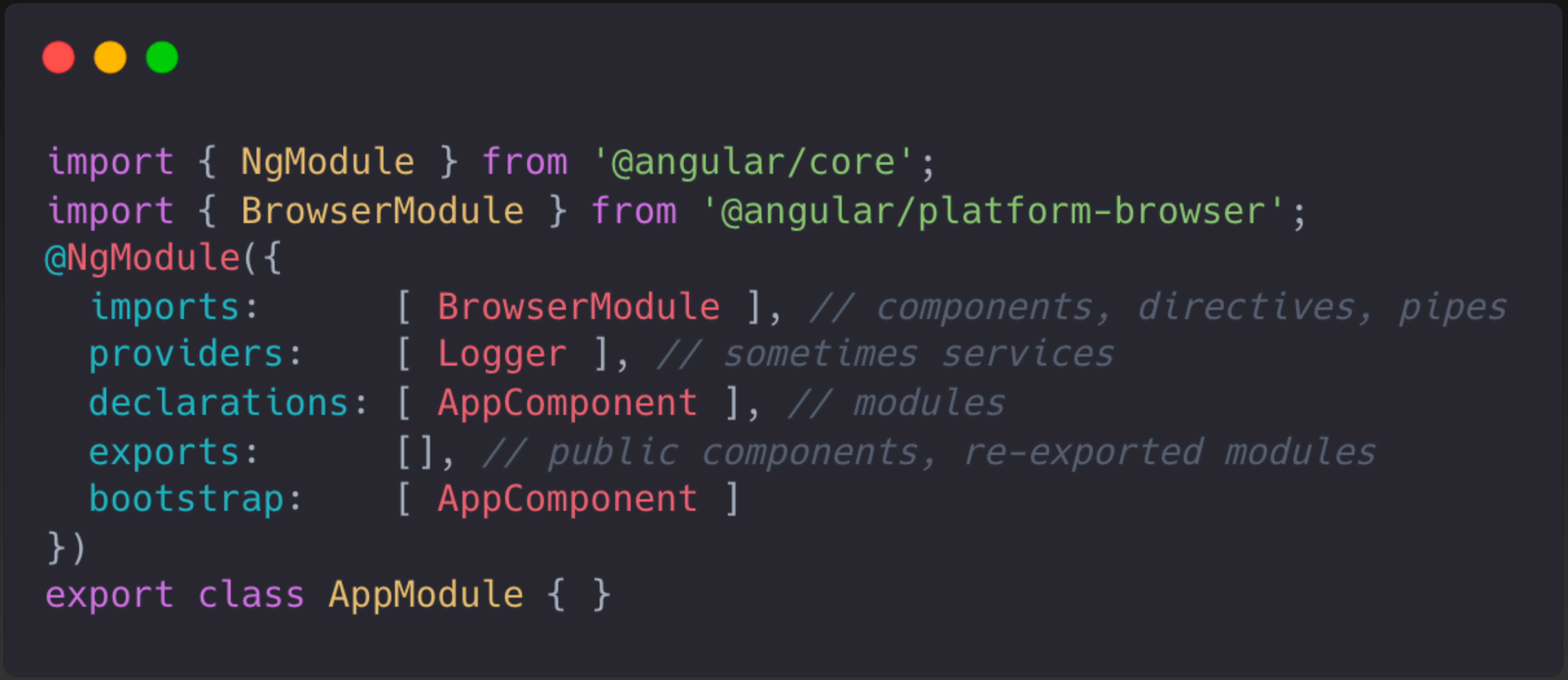
https://angular.io/cli

Architecture



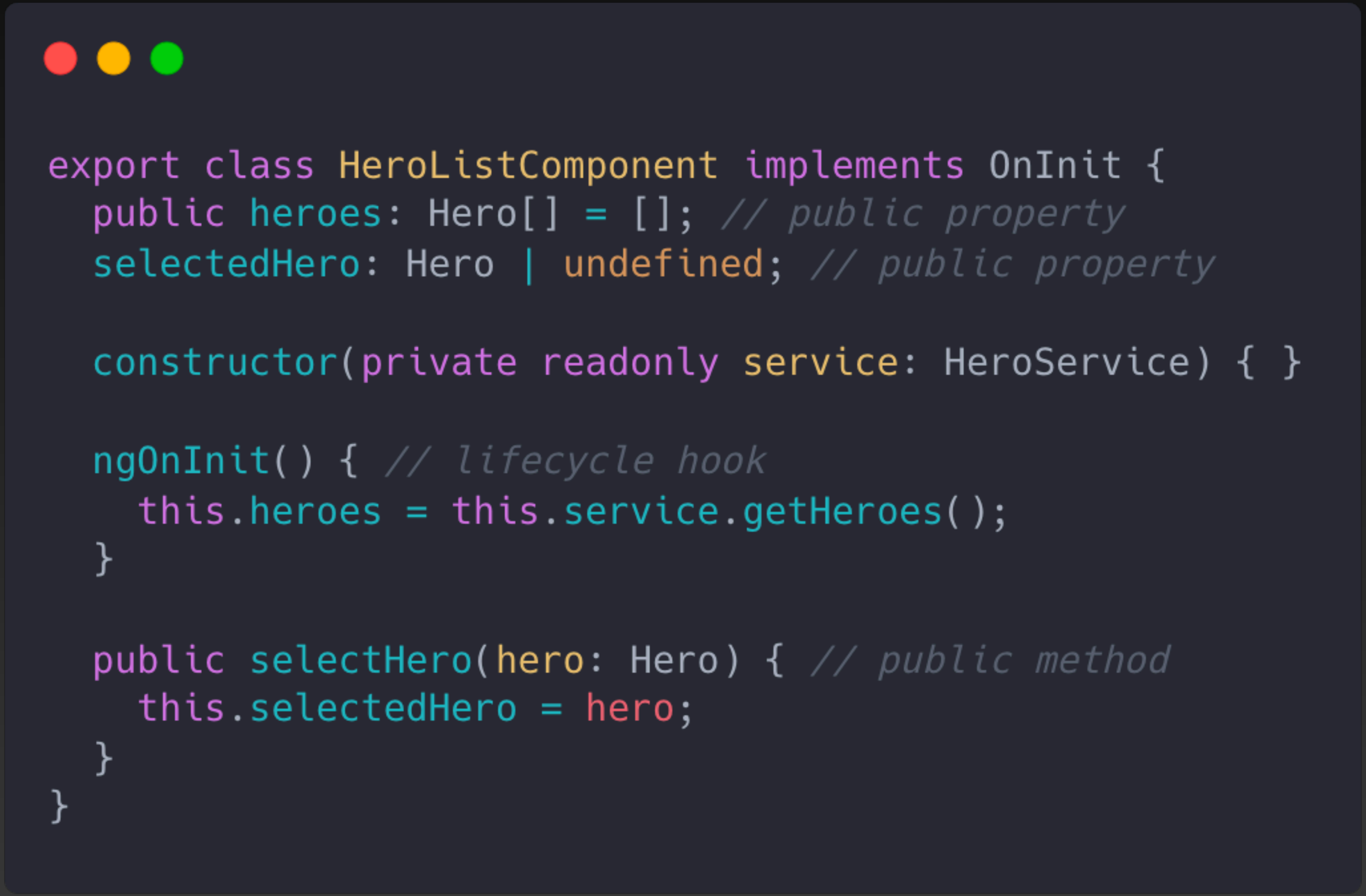
https://angular.io/guide/architecture

Modules



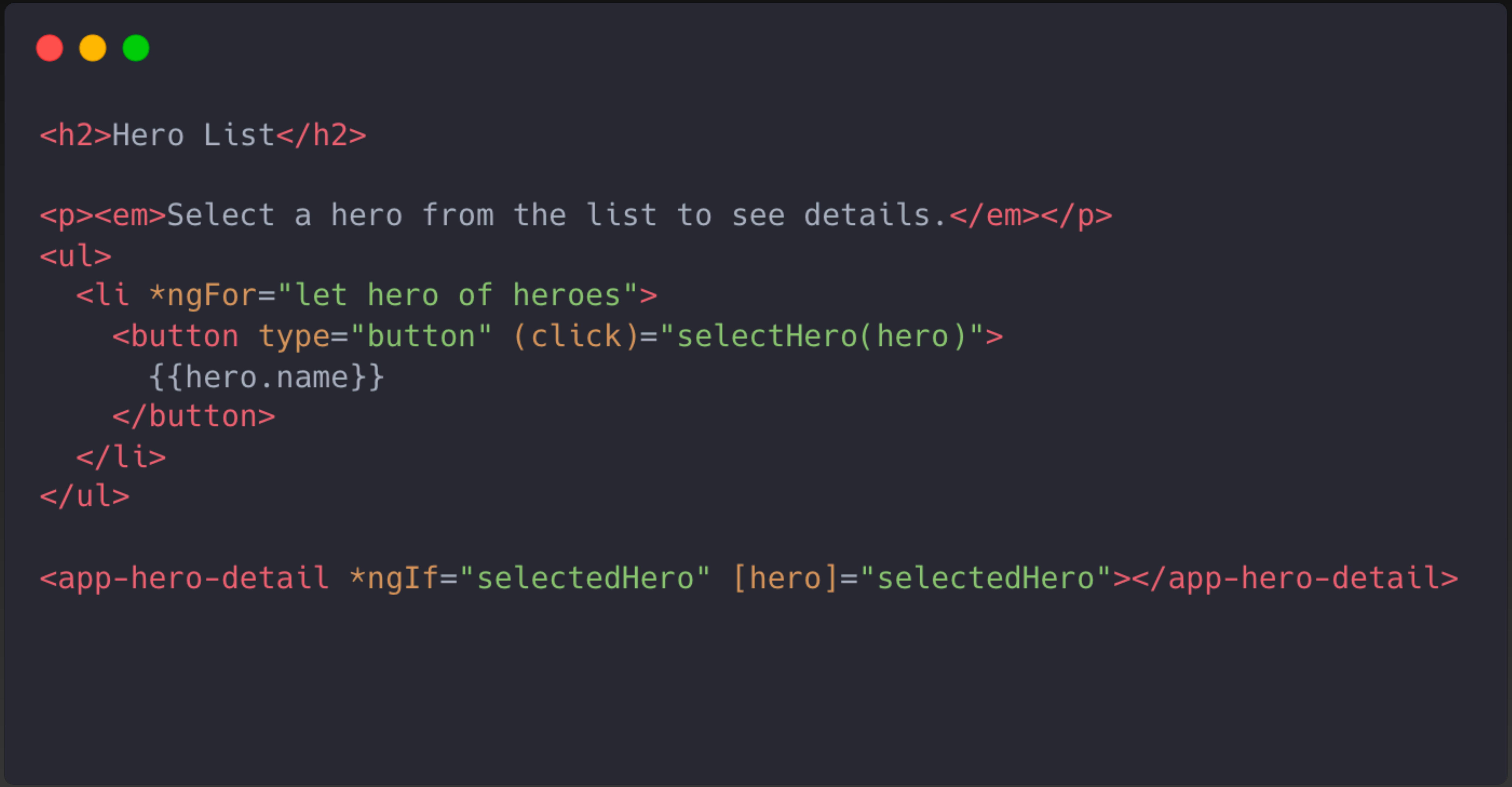
```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
@NgModule({
  imports:      [ BrowserModule ], // components, directives, pipes
  providers:    [ Logger ], // sometimes services
  declarations: [ AppComponent ], // modules
  exports:      [], // public components, re-exported modules
  bootstrap:    [ AppComponent ]
})
export class AppModule { }
```

Component class

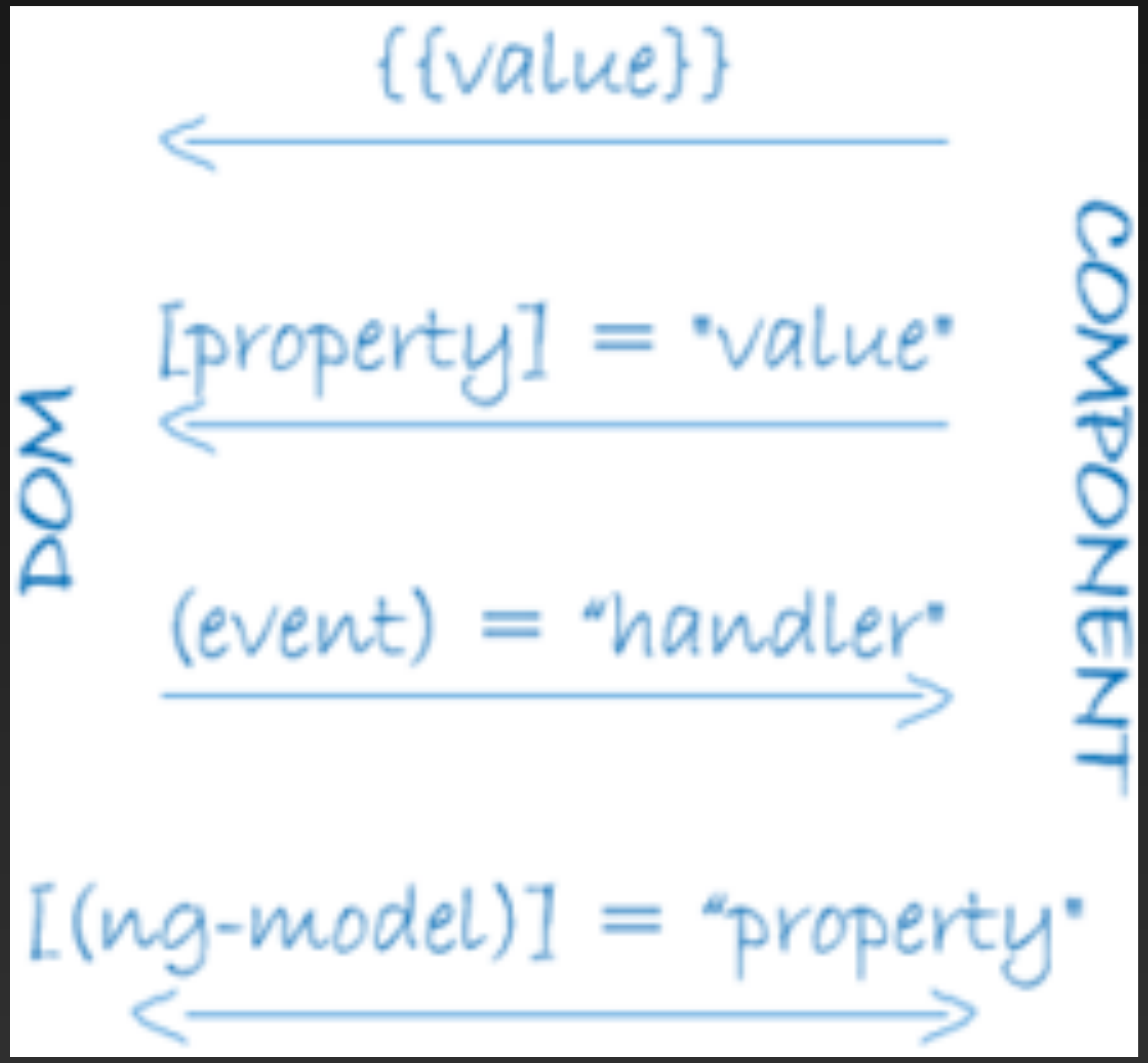


```
export class HeroListComponent implements OnInit {
  public heroes: Hero[] = []; // public property
  selectedHero: Hero | undefined; // public property

  constructor(private readonly service: HeroService) { }

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

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

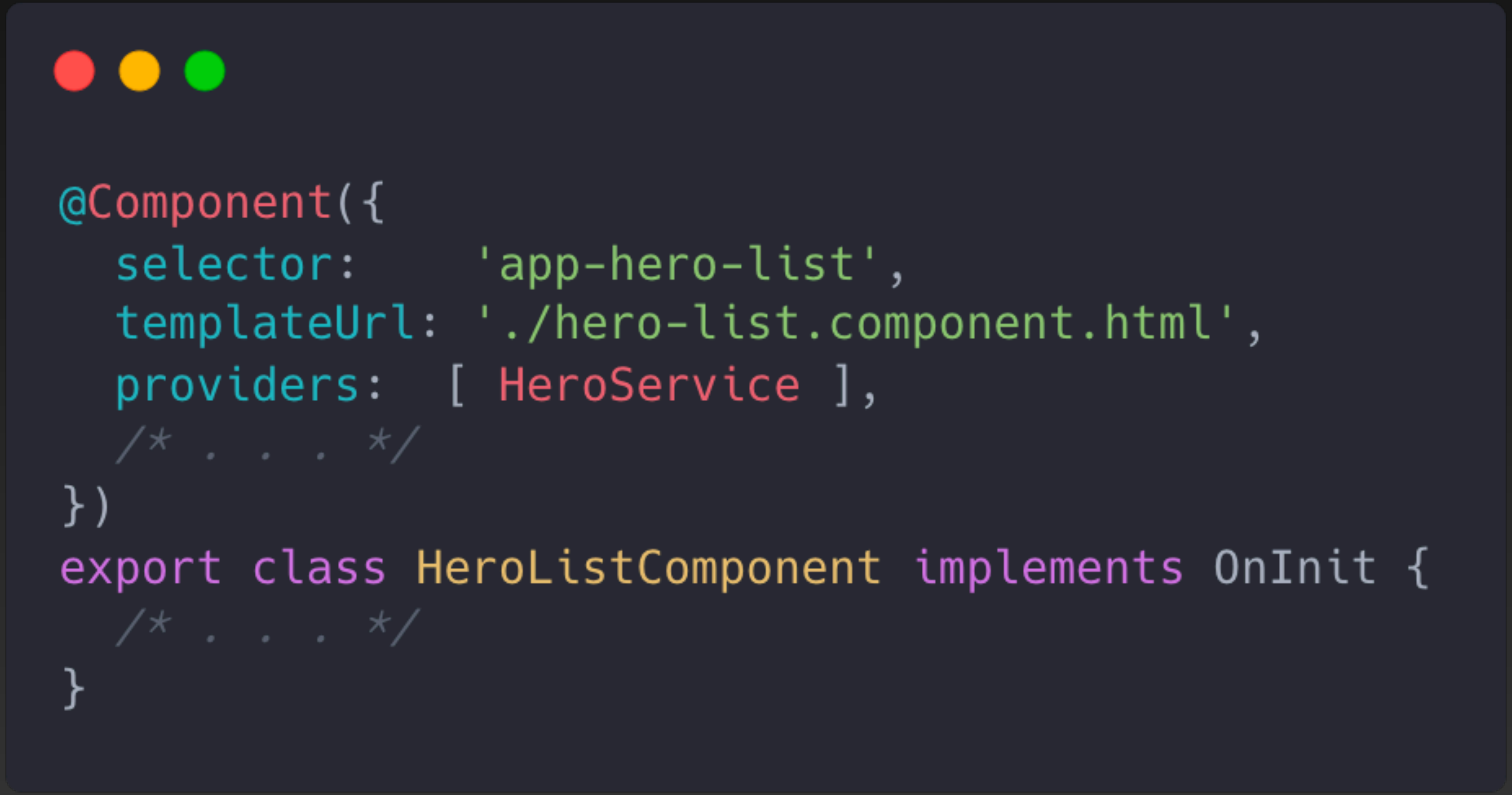
Component template



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

<p><em>Select a hero from the list to see details.</em></p>
<ul>
  <li *ngFor="let hero of heroes">
    <button type="button" (click)="selectHero(hero)">
      {{hero.name}}
    </button>
  </li>
</ul>

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



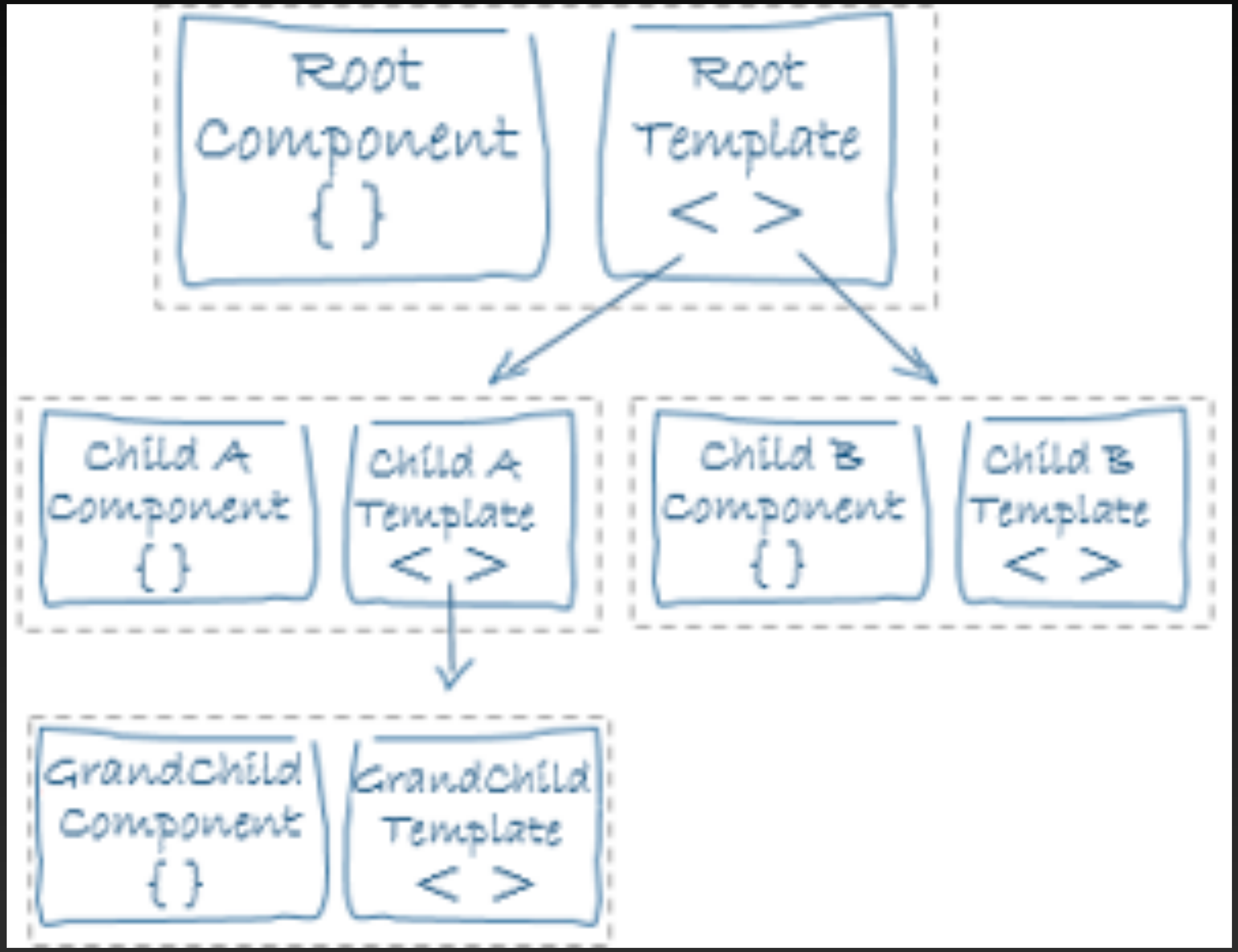
Component metadata



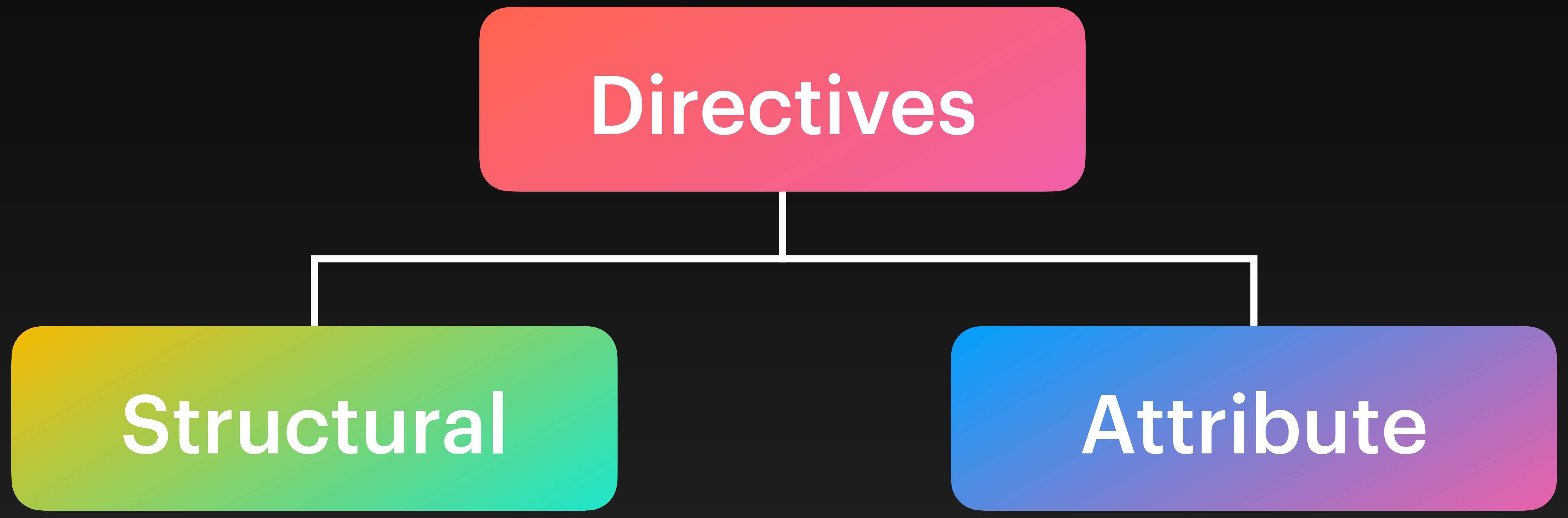
```
@Component({
  selector:    'app-hero-list',
  templateUrl: './hero-list.component.html',
  providers:  [ HeroService ],
  /* . . . */
})
export class HeroListComponent implements OnInit {
  /* . . . */
}
```

View hierarchy

- The template immediately associated with a component defines that component's host view. The component can also define a view hierarchy, which contains embedded views, hosted by other components.



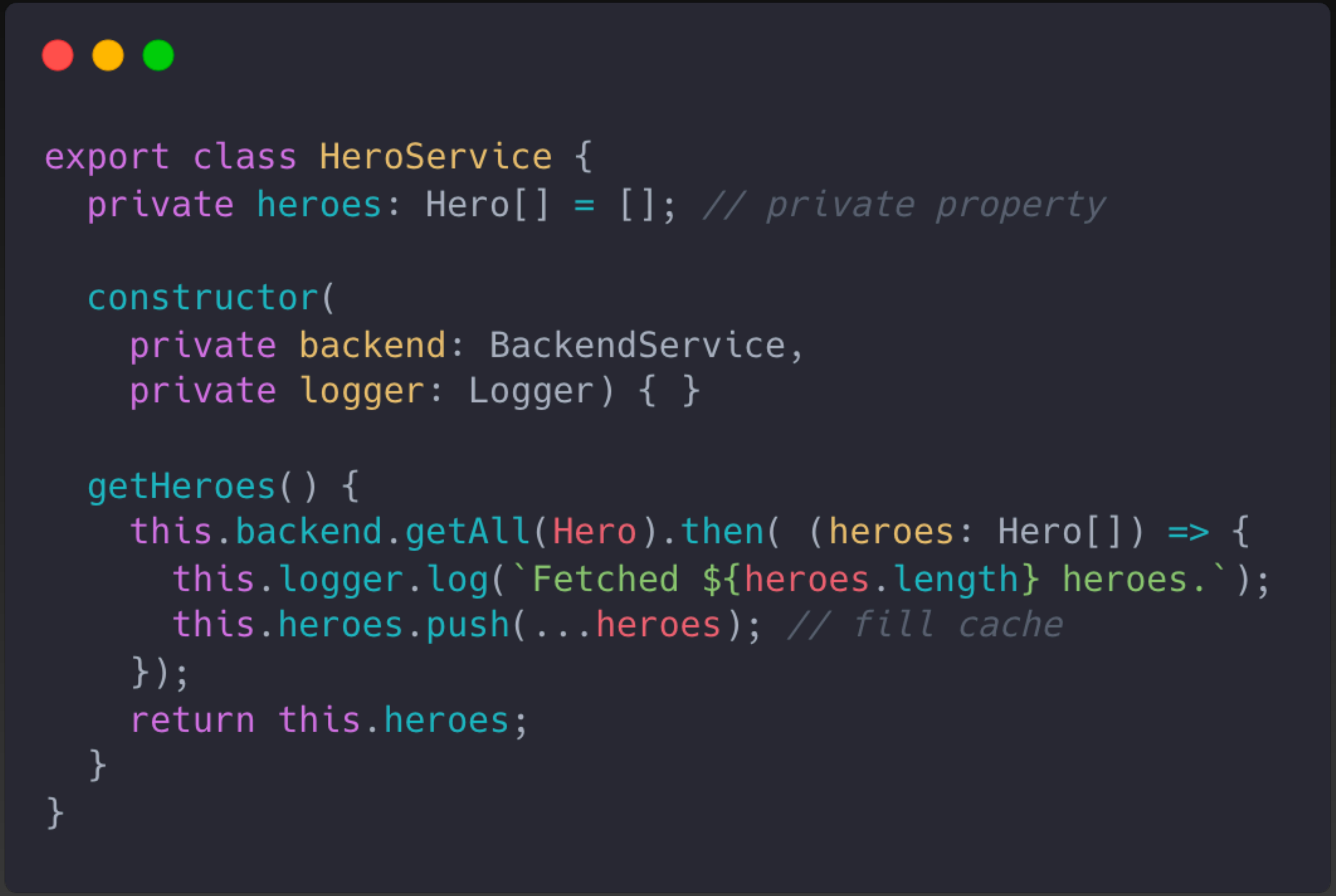
Built-in directives



- *ngFor
- *ngIf
- ngSwitch + *ngSwitchCase + *ngSwitchDefault

- ngModel
- ngStyle
- ngClass
- ngNonBindable

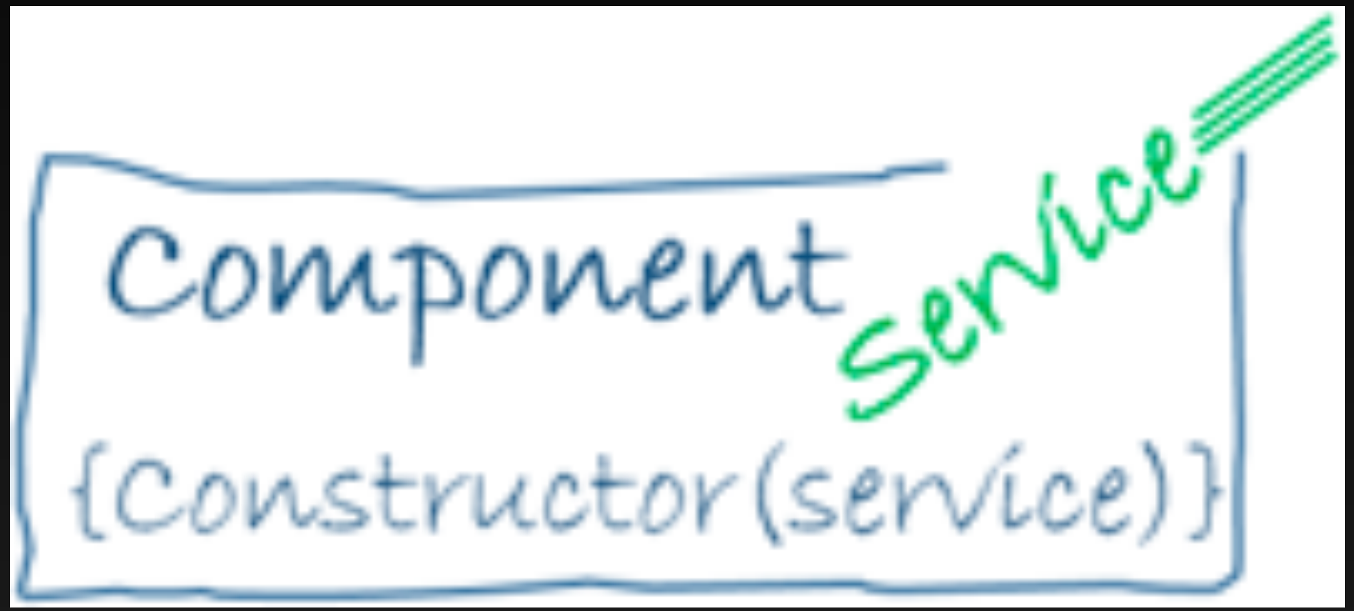
Service

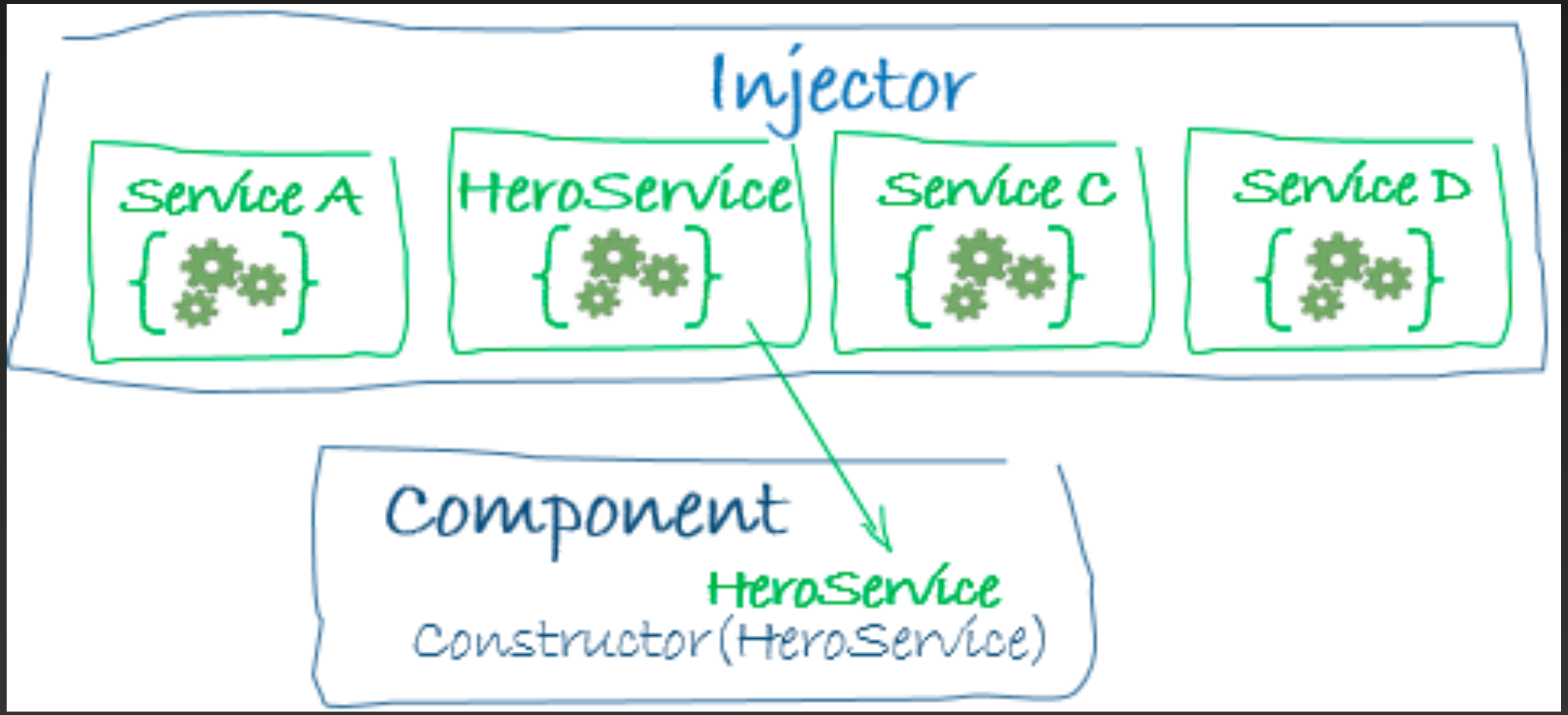


```
export class HeroService {
  private heroes: Hero[] = []; // private property

  constructor(
    private backend: BackendService,
    private logger: Logger) { }

  getHeroes() {
    this.backend.getAll(Hero).then( (heroes: Hero[]) => {
      this.logger.log(`Fetched ${heroes.length} heroes.`);
      this.heroes.push(...heroes); // fill cache
    });
    return this.heroes;
  }
}
```

Dependency injection

- Dependency injection (DI) is the part of the Angular framework that provides components with access to services and other resources. Angular provides the ability for you to inject a service into a component to give that component access to the service.





Useful links

- Official Angular documentation
  https://angular.io/docs
- In depth explanations about Angular
  https://indepth.dev/angular

Thank you!