

# Tecnologias e Programação Web

## 2019/2020

### Angular Framework



# Angular Framework

*Introduction to Angular*

# Angular Introduction (i)



- Angular Framework (current version – Angular 8 - 9) is a web application development platform based on TypeScript language.
- It makes the switch from MVC (Model-View-Controller) approach, used in AngularJS, to a Components-Based Web Development approach.

# Angular Introduction (ii)



- Angular allows to build web applications out of components, which are UI building blocks that are easy to test and reuse.
- Components, neatly encapsulate all the style and function required for a certain feature to work, and so, creating custom HTML elements is a greatly simplified process.

# Angular Introduction (iii)



- The idea is to build declarative components that are fully encapsulated.
- They describe their own views, and can be easily packaged and distributed to other developers.
- The application itself consists of a root component that contains a set of components for every UI element, screen, and route.
- This component tree lies at the core of any Angular application.



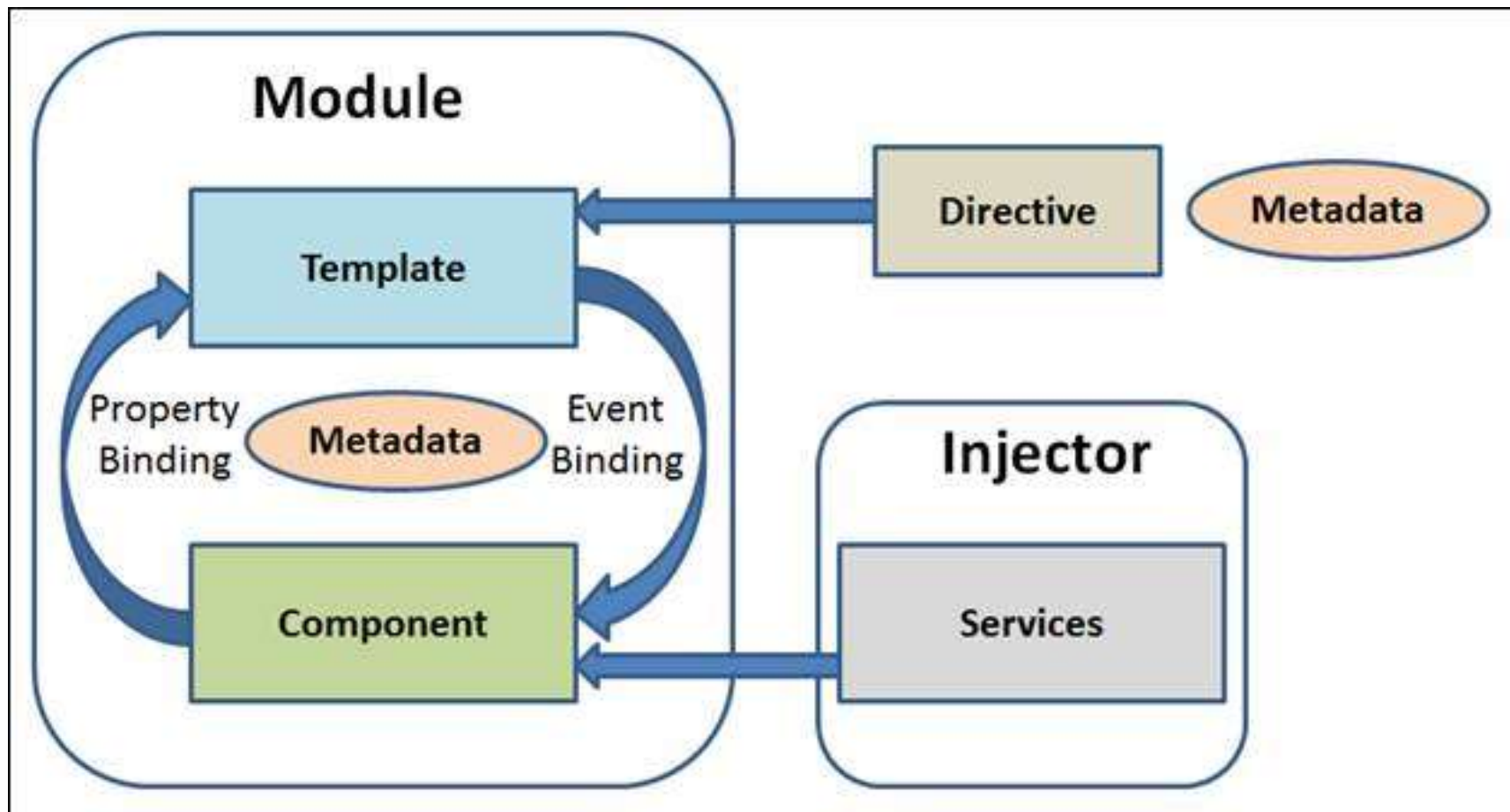
# Angular Framework

*Angular Architecture*

# Angular Architecture



- The main eight blocks of an Angular Application



# Angular – Modules (i)



- A Module is a block of code which is designed to perform a single task.
- It can be exported in form of class.
- Angular applications can have several modules. Although, every Angular application must have at least one.
- Every Angular application has one root module and can have many more featured modules.



# Angular – Modules (ii)



- An Angular module is a class which uses the `@NgModule` decorator.
- Decorators provide a way to add both annotations and a meta-programming syntax for class declarations and members.
- `@NgModule` takes a single metadata object and its properties to describe the module.

# Angular – Modules (iii)



- Following are the important properties of @NgModule:
  - exports – it's the subset of declarations which would be used in the component template of other module.
  - imports - imports other modules
  - providers - It is a creator of services. They can be accessible in all the parts of the application.
  - bootstrap - The root module has to set the bootstrap property. It is used to host all other views.
  - declarations - It declare the view class that belong to current module. There are three types of view classes supported by Angular: components, directives, and pipes.

# Angular – Components



- A component is the combination of a class, containing the core logic for a page, and an associated template that deals with its View.
- The application logic is written inside the class which is used by the View. The class interacts with the View through methods and properties of its API.
- Angular creates and updates the required components, and destroys the unused ones as the user moves through an application.

# Angular – Metadata



- Metadata is the way to define how Angular process a class, a method or a property, for particular reasons.
- In TypeScript, metadata is defined by using decorators.
- For example, if we want to define a component in an Angular application, we need to tell Angular that a particular class is a component.
  - This is done associating metadata to the class using `@Component` decorator.

# Angular – Template



- A template is the component View that tells Angular how to display the component.
- It looks like normal HTML.

# Angular – Data Binding



- Data binding is a powerful feature of software development technologies.
- It is the connection bridge between View and the business logic of the application.
- There are four types of data binding supported by Angular:
  - Interpolation - used to display the component value within HTML.
  - Property Binding - It passes the property's value of a parent component to its child's property.

# Angular – Data Binding



- Four types of data binding, continuation:
  - Event Binding – used to fire an event when we click on a component's name, or some change occurs in an input component.
  - Two-way Binding – it combines event and property binding in single notation by using ngModel directive, in order to have automatic change between the data model and the view.

# Angular – Directives



- Directives extend HTML attributes.
- These are markers on the DOM elements which provides some special behavior to them and tell Angular HTML compiler to attach it.
- There are three types of directives:
  - Decorator Directive - it decorates (@Directive) the elements using additional behavior. There are many built-in directives like ngModel, and others.
  - Component Directive - it is extended from @Directive decorator with template-oriented features.
  - Template Directive - it converts HTML into a reusable template. It is also known as structural directive.



# Angular – Service



- A service is a broad category encompassing any value, function, or feature that an app needs.
- A service is typically a class with a narrow, well-defined purpose.
- Angular distinguishes components from services in order to increase modularity and reusability.
- Typical examples of services are logging service, data service, message service etc. And there is no base class to define a service.

# Angular – Dependency Injection



- Dependency Injection is a software design pattern in which objects are passed as dependencies.
- It helps us remove the hard coded dependencies, and makes dependencies configurable.
- Using Dependency Injection, we can make components more maintainable, reusable, and testable.
- Dependency Injection (DI) is wired into the Angular framework and used everywhere to provide new components with the services or other things they need.



# Angular Framework

*Developing*

# Angular – Developing

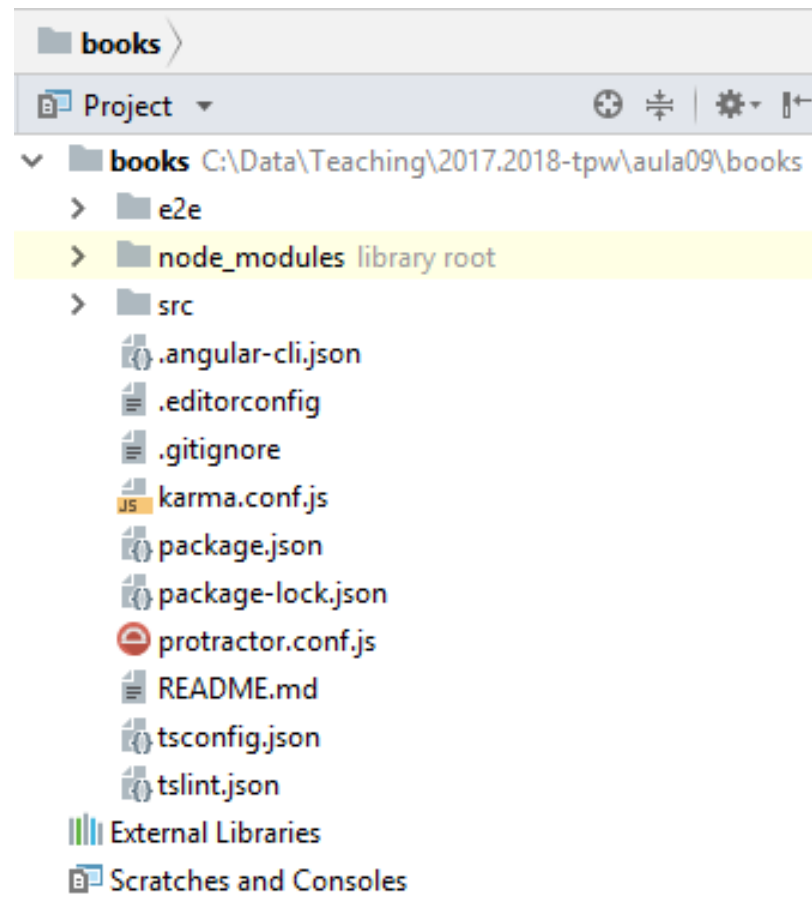


- Installation:
  - Install Node.js and npm if they are not already on your machine.
  - Install Angular CLI (Command Line Interface) globally.
    - command: `npm install -g @angular/cli`
- Creating a project
  - Use a know IDE, like WebStorm to create and develop a Angular project, or use the following command:
    - command: `ng new angular-app`
- Running the project
  - Use IDE to run it or use the following command, inside project folder:
    - command: `ng serve`

# Angular – Project Books



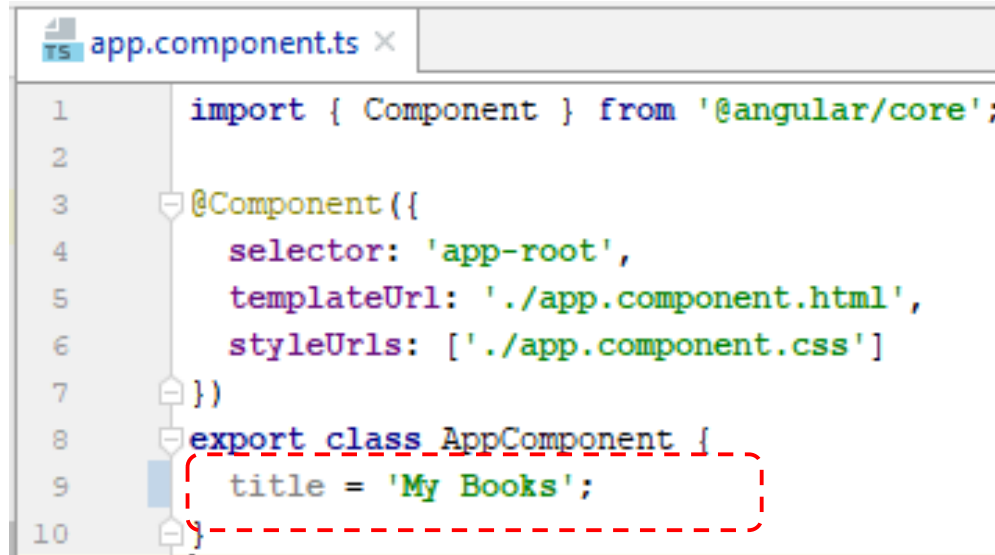
- Create a new Angular project, named “books”, using the command line or WebStorm IDE, load it and run it.



# Angular – Developing books (i)

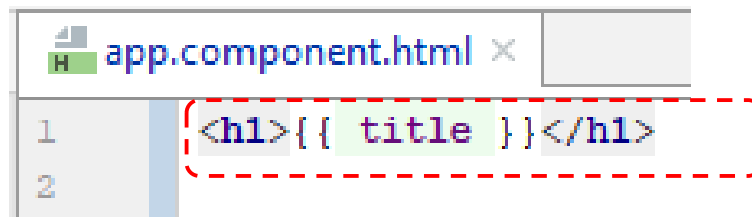


- Change title in main component.
- Open “src/app/app.component.ts” and change it.



```
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-root',
5   templateUrl: './app.component.html',
6   styleUrls: ['./app.component.css']
7 })
8 export class AppComponent {
9   title = 'My Books';
10 }
```

- Open “src/app/app.component.html” and change it.



```
1 <h1>{{ title }}</h1>
2
```

# Angular – Developing books (ii)



- Create application styles and save them on file “src/styles.css”.

```
h1 {  
  color: #369;  
  font-family: Arial, Helvetica, sans-serif;  
  font-size: 250%;  
}  
h2, h3 {  
  color: #444;  
  font-family: Arial, Helvetica, sans-serif;  
  font-weight: lighter;  
}  
body {  
  margin: 2em;  
}  
body, input[text], button {  
  color: #888;  
  font-family: Cambria, Georgia;  
}  
* {  
  font-family: Arial, Helvetica, sans-serif;  
}  
htz@ua.pt
```

# Angular – Developing books (iii)



- Create a component named “authors”, running the command inside project folder:  
“ng generate component authors”

```
authors.component.ts x
1  import { Component, OnInit } from '@angular/core';
2
3  @Component({
4    selector: 'app-authors',
5    templateUrl: './authors.component.html',
6    styleUrls: ['./authors.component.css']
7  })
8  export class AuthorsComponent implements OnInit {
9
10     constructor() { }
11
12     ngOnInit() {
13     }
14
15 }
```



# Angular – Developing books (iv)

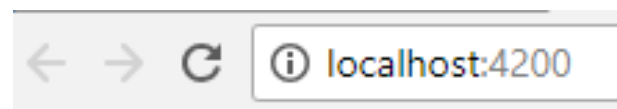


- Open “src/app/app.component.html” and change it.

A screenshot of a code editor showing the file 'app.component.html'. The code contains two lines: line 1 is '<h1>{{ title }}</h1>' and line 2 is '<app-authors></app-authors>'. A red dashed rectangle highlights both lines of code.

```
1 <h1>{{ title }}</h1>
2 <app-authors></app-authors>
3
```

- Result:



**My Books**

authors works!

# Angular – Developing books (v)



- Create Author class in file “src/app/author.ts”

```
1
2 export class Author {
3     num: number;
4     name: string;
5     email: string;
6 }
```

# Angular – Developing books (vi)



- Create authors list file “src/app/authorslist.ts”

```
authorslist.ts x
1
2  import {Author} from "../author";
3
4  export const AUTHORS: Author[] = [
5    { num: 1, name: 'Fernando Pessoa', email: 'fpessoa@mail.pt' },
6    { num: 2, name: 'J. K. Rowling', email: 'jkrowling@mail.uk' },
7    { num: 3, name: 'Stephen King', email: 'sking@aol.com' },
8    { num: 4, name: 'Arthur Conan Doyle', email: 'acdoyle@mail.net' },
9    { num: 5, name: 'John Green', email: 'jgreen@mail.com' },
10  ];
```

# Angular – Developing books (vii)



- Change “src/app/authors/authors.component.ts” file:

```
1  import { Component, OnInit } from '@angular/core';
2  import { Author } from "../author";
3  import { AUTHORS } from "../authorslist";
4
5  @Component({
6    selector: 'app-authors',
7    templateUrl: './authors.component.html',
8    styleUrls: ['./authors.component.css']
9  })
10 export class AuthorsComponent implements OnInit {
11
12    authors: Author[];
13
14    constructor() {
15      this.authors = AUTHORS;
16    }
17
18    ngOnInit() {
19    }
20
21 }
```

# Angular – Developing books (viii)



- Change “src/app/authors/authors.component.html” file:

```
1
2 <h2>Authors</h2>
3 <ul class="authors">
4   <li *ngFor="let author of authors">
5     <span class="badge">{{ author.num }}</span> {{ author.name }}
6   </li>
7 </ul>
```

# Angular – Developing books (ix)



- Create component styles and save them on file “src/app/authors/authors.component.css”.

```
/* AuthorsComponent's private CSS styles */
.selected {
  background-color: #CFD8DC !important;
  color: white;
}
.authors {
  margin: 0 0 2em 0;
  list-style-type: none;
  padding: 0;
  width: 15em;
}
.authors li {
  cursor: pointer;
  position: relative;
  left: 0;
  background-color: #EEE;
  margin: .5em;
  padding: .3em 0;
  height: 1.6em;
  border-radius: 4px;
}
...
```

# Angular – Developing books (x)

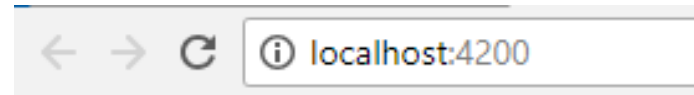


```
...  
.authors li.selected:hover {  
  background-color: #BBD8DC !important;  
  color: white;  
}  
.authors li:hover {  
  color: #607D8B;  
  background-color: #DDD;  
  left: .1em;  
}  
.authors .text {  
  position: relative;  
  top: -3px;  
}  
.authors .badge {  
  display: inline-block;  
  font-size: small;  
  color: white;  
  padding: 0.8em 0.7em 0 0.7em;  
  background-color: #607D8B;  
  line-height: 1em;  
  position: relative;  
  left: -1px;  
  top: -4px;  
  height: 1.8em;  
  margin-right: .8em;  
  border-radius: 4px 0 0 4px;  
}
```

# Angular – Developing books (xi)



- Result:



## My Books

### Authors

1	Fernando Pessoa
2	J. K. Rowling
3	Stephen King
4	Arthur Conan Doyle
5	John Green



# Angular – Developing books (xii)



- Change “src/app/authors/authors.component.html” file:

```
authors.component.html x
1
2 <h2>Authors</h2>
3 <ul class="authors">
4   <li *ngFor="let author of authors" (click)="onSelect(author)">
5     <span class="badge">{{ author.num }}</span> {{ author.name }}
6   </li>
7 </ul>
```

# Angular – Developing books (xiii)



- Change “src/app/authors/authors.component.ts” file:

```
authors.component.ts x
1  import { Component, OnInit } from '@angular/core';
2  import { Author } from "../author";
3  import { AUTHORS } from "../authorslist";
4
5  @Component({
6    selector: 'app-authors',
7    templateUrl: './authors.component.html',
8    styleUrls: ['./authors.component.css']
9  })
10 export class AuthorsComponent implements OnInit {
11
12    authors: Author[];
13    selectedAuthor: Author;
14
15    constructor() {
16      this.authors = AUTHORS;
17    }
18
19    onSelect(author: Author): void {
20      this.selectedAuthor = author;
21    }
22  }
```

# Angular – Developing books (xiv)



- Change “src/app/authors/authors.component.html” file:

```
authors.component.html x
2      <h2>Authors</h2>
3      <ul class="authors">
4        <li *ngFor="let author of authors" (click)="onSelect(author)">
5          <span class="badge">{{ author.num }}</span> {{ author.name }}
6        </li>
7      </ul>
8
9      <div *ngIf="selectedAuthor">
10        <h2>Information on {{ selectedAuthor.name | uppercase }} </h2>
11        <div><span>Num: </span>{{selectedAuthor.num}}</div>
12        <div>
13          <label>Name:
14            <input [(ngModel)]="selectedAuthor.name" placeholder="name">
15          </label>
16        </div>
17        <div>
18          <label>Email:
19            <input [(ngModel)]="selectedAuthor.email" placeholder="email">
20          </label>
21        </div>
22      </div>
```

# Angular – Developing books (xv)



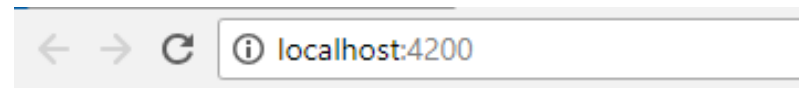
- Change “src/app/app.module.ts” file:

```
app.module.ts x
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3  import { FormsModule } from "@angular/forms";
4
5  import { AppComponent } from './app.component';
6  import { BooksComponent } from './books/books.component';
7  import { AuthorsComponent } from './authors/authors.component';
8
9
10 @NgModule({
11   declarations: [
12     AppComponent,
13     BooksComponent,
14     AuthorsComponent
15   ],
16   imports: [
17     BrowserModule,
18     FormsModule
19   ],
20   providers: [],
21   bootstrap: [AppComponent]
22 })
```

# Angular – Developing books (xvi)



- Result:



## My Books

### Authors

1	Fernando P
2	J. K. Rowling
3	Stephen King
4	Arthur Conan Doyle
5	John Green

### Information on FERNANDO P

Num: 1

Name:

Email: