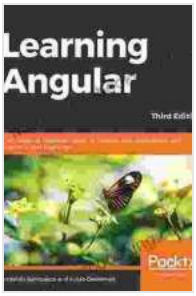
No-Nonsense Beginner's Guide to Building Web Applications with Angular 10

In today's fast-paced digital world, web applications are an essential part of our daily lives. Whether it's for work, play, or anything in between, we rely on web apps to connect with others, access information, and complete tasks. If you're interested in learning how to build your own web applications, Angular 10 is an excellent framework to start with.

Angular is a powerful and versatile open-source framework that makes it easy to build scalable, maintainable, and performant web applications. It's used by some of the world's largest companies, including Google, Microsoft, and Our Book Library.



Learning Angular: A no-nonsense beginner's guide to building web applications with Angular 10 and TypeScript, 3rd Edition by Aristeidis Bampakos

4.1 out of 5

Language	: English
File size	: 4989 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Print length	: 430 pages

In this beginner's guide, we'll walk you through everything you need to know to get started with Angular 10, from setting up your development

environment to creating your first web application. We'll cover all the basics, including components, directives, services, and routing.

By the end of this guide, you'll have a solid foundation in Angular 10 and be able to start building your own web applications.

Getting Started

To get started with Angular 10, you'll need to install the following software:

- Node.js (version 10 or higher)
- Angular CLI (version 10 or higher)

Once you have installed the required software, you can create a new Angular project by running the following command in your terminal:

ng new my-project

This command will create a new directory called `my-project` and install all of the necessary dependencies.

Once the project has been created, you can navigate to the project directory and start the development server by running the following command:

ng serve

The development server will run on port 4200 by default. You can access your application by visiting `http://localhost:4200` in your browser.

Components

Components are the basic building blocks of Angular applications. They are reusable, self-contained units of code that define the structure and behavior of a specific part of your application.

Components are typically composed of three parts:

- **Template**: The template defines the HTML layout of the component.
- **Class**: The class defines the behavior of the component, such as its data and methods.
- **Styles**: The styles define the appearance of the component, such as its colors and fonts.

Here's an example of a simple Angular component:

@Component({ selector: 'my-component', template: '' }) export class MyComponent {}

This component defines a new HTML element called `<my-component>` that displays the text "Hello, world!".

Directives

Directives are used to modify the behavior of elements in your application. They can be used to add new functionality to elements, such as event handling or data binding.

There are two types of directives:

- **Attribute directives**: Attribute directives are applied to elements using attribute syntax.

- **Structural directives**: Structural directives are applied to elements using asterisks (*).

Here's an example of an attribute directive that adds a click event handler to an element:

@Directive({ selector: '[myClick]' }) export class MyClickDirective { @HostListener('click') onClick(){console.log('Clicked!'); }}

This directive can be applied to any element using the `myClick` attribute. When the element is clicked, the `onClick()` method will be called.

Services

Services are used to share data and functionality between different parts of your application. They can be used to store data, perform calculations, or communicate with external APIs.

To create a service, you can use the `@Injectable()` decorator.

@Injectable({ providedIn: 'root' }) export class MyService { getData(){return ['foo', 'bar', 'baz']; }}

This service defines a method called `getData()` that returns an array of data.

To use the service in your component, you can inject it into the constructor.

export class MyComponent { constructor(private myService: MyService) {}

ngOnInit(){this.data = myService.getData(); }}

Routing

Routing is used to navigate between different pages in your application.

To set up routing in your application, you can use the `@Routes()` decorator.

@Routes([{ path: '', component: HomeComponent },{path: 'about', component: AboutComponent }]) export class MyAppComponent {}

This decorator defines two routes: one for the home page and one for the about page.

To navigate between routes, you can use the `Router` service.

export class MyComponent { constructor(private router: Router) {}

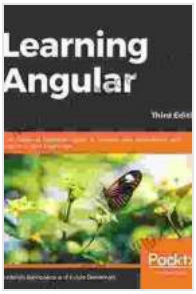
goToAboutPage(){this.router.navigate(['/about']); }}

This is just a brief overview of the basics of Angular 10. For more in-depth information, I recommend checking out the official Angular documentation.

If you're looking to learn more about Angular 10, I also recommend checking out the following resources:

- Angular Tutorial
- Angular Documentation
- Angular Fundamentals Course

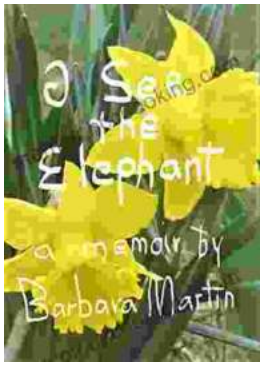
With a little effort, you'll be able to master Angular 10 and build amazing web applications.



Learning Angular: A no-nonsense beginner's guide to building web applications with Angular 10 and TypeScript, 3rd Edition by Aristeidis Bampakos

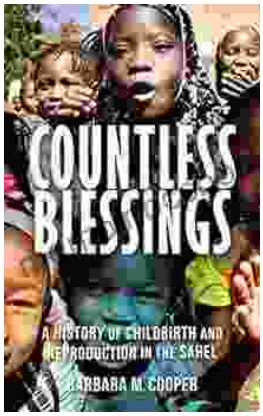
4.1 out of 5

Language	: English
File size	: 4989 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Print length	: 430 pages



Teach Your Child They Have No Self Worth And They Will Live An Unfulfilled Life

By Dr. Jane Doe As a parent, you want what is best for your child. You want them to be happy, healthy, and successful. However, there are some...



Unveiling Centuries of Tradition: History of Childbirth and Reproduction in the Sahel

Journey into the heart of the Sahel, a vast and enigmatic region where childbirth and reproduction have played a pivotal role in shaping human history. "History of...