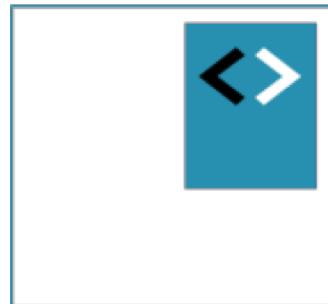


Rangle's Angular Training Book

brief reading guide



A CANON COMPANY



Peter Kassenaar –
info@kassenaar.com

<https://angular-2-training-book.rangle.io/>

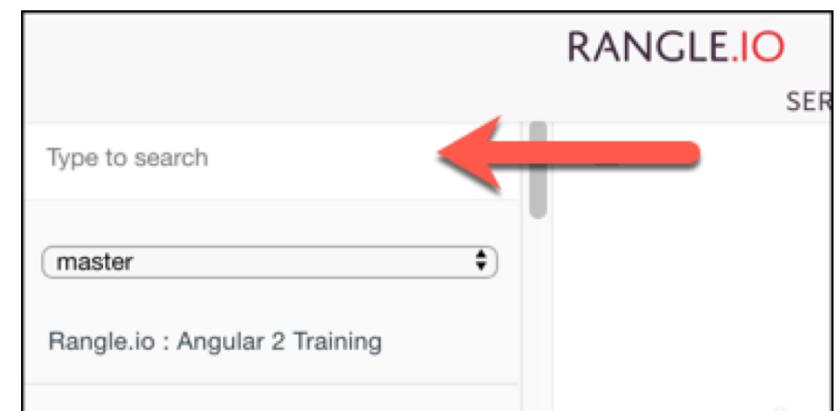
Rangle's Angular Training Book



Over the last three and a half years, Angular has become the leading open source JavaScript application framework for hundreds of thousands of programmers around the world. The "1.x" version of Angular has been widely used and became extremely popular for complex applications. The new Angular 2.x has also announced its final release version.

In General

- In addition to the Dutch book on Angular
- The book can be read linearly.
- Each page has previous/next indicators on the sides.
- Not everything is important for the Océ Angular Fundamentals training.
- The order in this book, differs from the training.
- You can fulltext-search the book!



Chapters

- **EcmaScript 6 and TypeScript Features** – Read if you want to know more about the backgrounds of ES6 and TypeScript. No 100% knowledge required for the training.

The slide has a sidebar on the left containing a list of topics:

- Creating Functional Forms
- EcmaScript 6 and TypeScript Features** (highlighted with a red circle)
- ES6
 - Classes
 - Refresher on 'this'
 - Arrow Functions
 - Template Strings
 - Inheritance
 - Delegation
 - Constants and Block Scoped ...
 - ...spread and ...rest
 - Destructuring
 - Modules
 - TypeScript
 - Getting Started With TypeScript
- Typings
- Linting

The main content area is titled "EcmaScript 6 and TypeScript Features" and features a graphic with "JS" and "ES6" on a yellow background and "TypeScript" on a blue background.

Figure: ES6 and TypeScript

The language we usually call "JavaScript" is formally known as "EcmaScript". The new version of JavaScript, known as "ES6", offers a number of new features that extend the power of the language.

ES6 is not widely supported in today's browsers, so it needs to be transpiled to ES5. You can choose between several transpilers, but we'll be using TypeScript, which is what the Angular team uses to write Angular. Angular makes use of a number of features of ES6 and TypeScript.

Chapters

- **The JavaScript Toolchain** – Useful background information on Git, Command Line and more. Recommended.

[The JavaScript Toolchain](#)

Source Control: git

The Command Line

Command Line JavaScript: NodeJS

Back-End Code Sharing and Dist...

Module Loading, Bundling and B...

Chrome

Bootstrapping an Angular Application

Understanding the File Structure

Bootstrapping Providers

Components in Angular

Creating Components

Application Structure with Comp...

The JavaScript Toolchain

In this section, we'll describe the tools that you'll be using for the rest of the course.



A photograph showing a variety of hand tools, such as wrenches, pliers, and screwdrivers, all featuring red handles. They are arranged on a light-colored wooden surface. A cursor icon is visible on the left side of the slide, pointing towards the sidebar.

Chapters

- **Bootstrapping an Angular Application** – Useful background information on Git, Command Line and more. Recommended.
- **Understanding the File Structure** – Important!
- **Bootstrapping Providers** – less important in the beginning. Read this again later, if you know more about Services. Then it all falls into place.

Chapters

- **Components in Angular**– Comparison between AngularJS 1.x directives and Angular 2 components. Can be useful.
- Read this chapter. Especially as a reference. You can forget about *Content Projection* for the moment.

Components in Angular

- Creating Components
- Application Structure with Comp...
- Passing Data into a Component
- Responding to Component Ev...
- Using Two-Way Data Binding
- Accessing Child Components ...
- Projection
- Structuring Applications with Co...
- Using Other Components

Directives

- Attribute Directives
- NgStyle Directive
- NgClass Directive
- Structural Directives
- Nglf Directive

Components in Angular

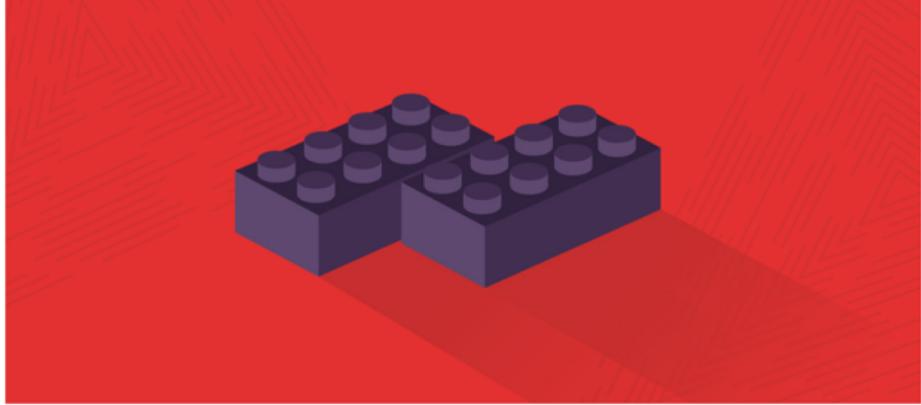


Figure: components

The core concept of any Angular application is the *component*. In effect, the whole application can be modeled as a tree of these components.

Chapters

- **Directives** – Good as a reference. Background info on `*ngFor` and `*ngIf`
- `NgSwitch`, `NgStyle` and `NgClass` are used less frequently.
- **Advanced Components** - Interesting as a reference. `ViewChild()` and View Encapsulation were not discussed in training, but possibly interesting for you.
- **Observables** - Important chapter. Discussed differently than in the training, but certainly a good reference.

Chapters

- **Angular Dependency Injection** – Background information on using DI. Particular attention to mechanics. In the training, the emphasis is on *using* DI, not on the internal operation.
- **HTTP** – Extensive reference on working with `Http` requests. However, in the 'old' way, by using the `HttpModule`. We now use `HttpClientModule`.
- At the time of writing, the book had not yet been adapted to this.
- Interesting introduction to other RxJS concepts, but not 100% important at this moment.
 - We now use the `.pipe()` operator in RxJS

Chapters

- **Change Detection** – Good as a reference. Not required to know the internals of CD. Only if you are interested.
- **Zones** - Idem.
- **Advanced Angular** – Idem
- **AoT in Angular** – Idem

Chapters

- **Immutable.js** – Again: good to get to know the internals of this.
- At the moment, however, it is used little as a stand-alone principle.
Often in combination with State Management tool like `@ngrx/store`.
- *Not mandatory!*
- **Pipes** – useful background information. The successor of Angular 1.x Filters.
- **Forms** – Interesting chapter with background information on Forms.
Reading is recommended.

Chapters

- **Modules** – Useful to learn more about dividing larger Angular Applications into separate feature modules. Recommended. We will cover this in the *Advanced* training.
- **Routing** – Read this. At least the basics. Not everything is equally important, but routing is an indispensable part of almost every Angular web application.

The screenshot shows a sidebar on the left with a list of topics under 'Sharing the Same Dependency In...' and 'Routing'. The 'Why Routing?' topic is highlighted in blue. The main content area has a title 'Why Routing?' and a detailed explanation of what routing does and why it's useful. It also lists what routing allows you to do and provides a bulleted list of benefits.

Sharing the Same Dependency In...

Routing

[Why Routing?](#)

Configuring Routes

Redirecting the Router to Another...

Defining Links Between Routes

Dynamically Adding Route Comp...

Using Route Parameters

Defining Child Routes

Controlling Access to or from a R...

Passing Optional Parameters to ...

Why Routing?

Routing allows us to express some aspects of the application's state in the URL. Unlike with server-side front-end solutions, this is optional - we can build the full application without ever changing the URL. Adding routing, however, allows the user to go straight into certain aspects of the application. This is very convenient as it can keep your application linkable and bookmarkable and allow users to share links with others.

Routing allows you to:

- Maintain the state of the application
- Implement modular applications
- Implement the application based on the roles (certain roles have access to certain URLs)

Chapters

- **State Management** – for the advanced
- **TDD Testing** – Recommended. If you want to know more about TDD and unit testing.
- **Migrating Angular 1.x Projects to Angular 2** – Not of interest for Océ.
- **Project Setup** – Not important if you use Angular CLI (like we do). You don't have to configure WebPack manually.

Chapters

- **Angular CLI** – Read this
- **Web Accessibility in Angular** – Optional. If you want to know more about Accessibility of your applications for disabled users.
- **Internationalization in Angular** – Recommended when creating multiple-language apps. Otherwise not.
- **Further Reading and Reference** – Useful list with pointers to more information.

Further Reading and Reference

Angular

- [Angular.io API Reference](#) - Angular Reference Material with easy access to different Angular items
- [Angular Style Guide](#) - Opinions from the Angular team
- [Angular Module Github](#) - Source code is written in readable TypeScript
- [Angular Material Github](#) - Official repo for Angular implementation in material design

TypeScript

- [tsconfig options](#) - information on how to configure the TypeScript compiler
- [TypeScript Playground](#) - In-browser TypeScript editor with live reload
- [TypeScript Handbook](#)
- [TypeScript Deep Dive](#) - Additional learning material

General Coding Practice and Functional Programming

<https://angular-2-training-book.rangle.io/handout/further-reading.html>

Questions, additions, remarks?

Let me know, via
info@kassenaar.com