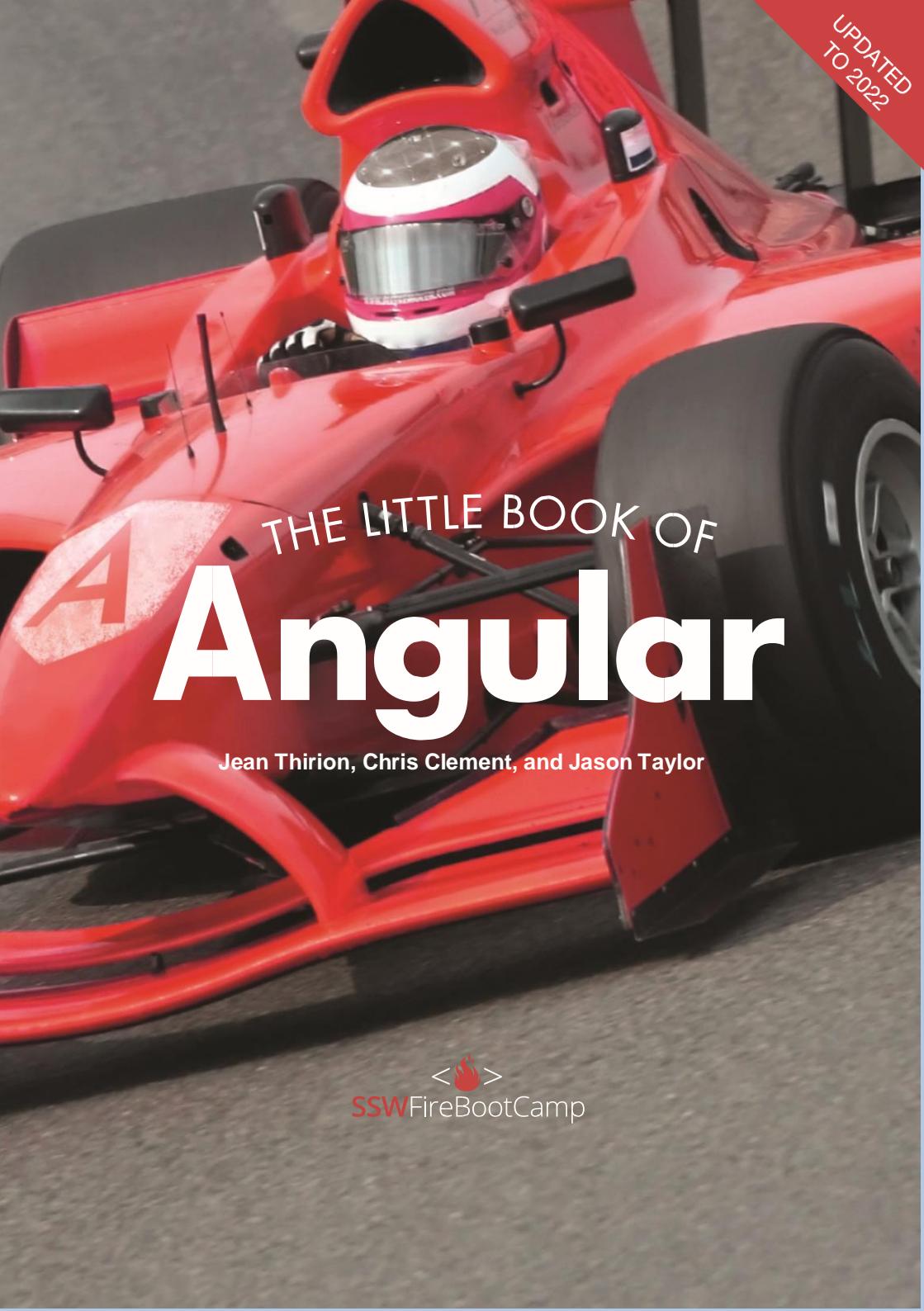


UPDATED
TO 2022

A dynamic photograph of a bright red Formula 1 racing car in motion. The car's front wheel and suspension are visible on the right, while the cockpit and front wing are on the left. A driver wearing a white and pink helmet is seated in the cockpit. The background is blurred, suggesting speed.

THE LITTLE BOOK OF Angular

Jean Thirion, Chris Clement, and Jason Taylor

<🔥>
SSWFireBootCamp

Why Angular is Awesome

Why you should use Angular



Angular is the best development platform for building mobile and desktop web applications

Angular is Fast

Angular is Cross-Platform

Code generation

Turn your templates into code optimized for today's JavaScript virtual machines, giving you the benefits of hand-written code with the productivity of a framework.

Universal

Serve the first view of your application on node.js .NET, PHP, and other servers for near-instant rendering in just HTML and CSS. This also paves the way for sites that optimize for SEO.

Code Splitting

Load quickly with the new Component Router, which delivers automatic code-splitting, so users only load code required to render the view they request.

Progressive Web Apps

Use modern web platform capabilities to deliver app-like experiences, with features such as high performance, offline and zero-step installation.

Native

Avoid development of a native mobile app by building an Angular application for Desktop, IOS and Android.

Desktop

Create desktop-installed apps across Mac, Windows and Linux using the same methods you've learned for the web, plus the ability to access native OS APIs.

Be more productive

Faster Development Lifecycle

Templates

Quickly create UI views with simple and powerful template syntax.

Angular CLI

Command line tools: start building fast, add components and tests, then instantly deploy.

IDEs

With TypeScript, you get intelligent code completion, instant errors and other feedback in popular editors and IDEs.

Testing

With Karma for unit tests, you can know if you've broken things every time you make changes, resulting in faster and more reliable development experience. From Jest to Cypress.io, there are so many rich testing libraries to choose from.

Animation

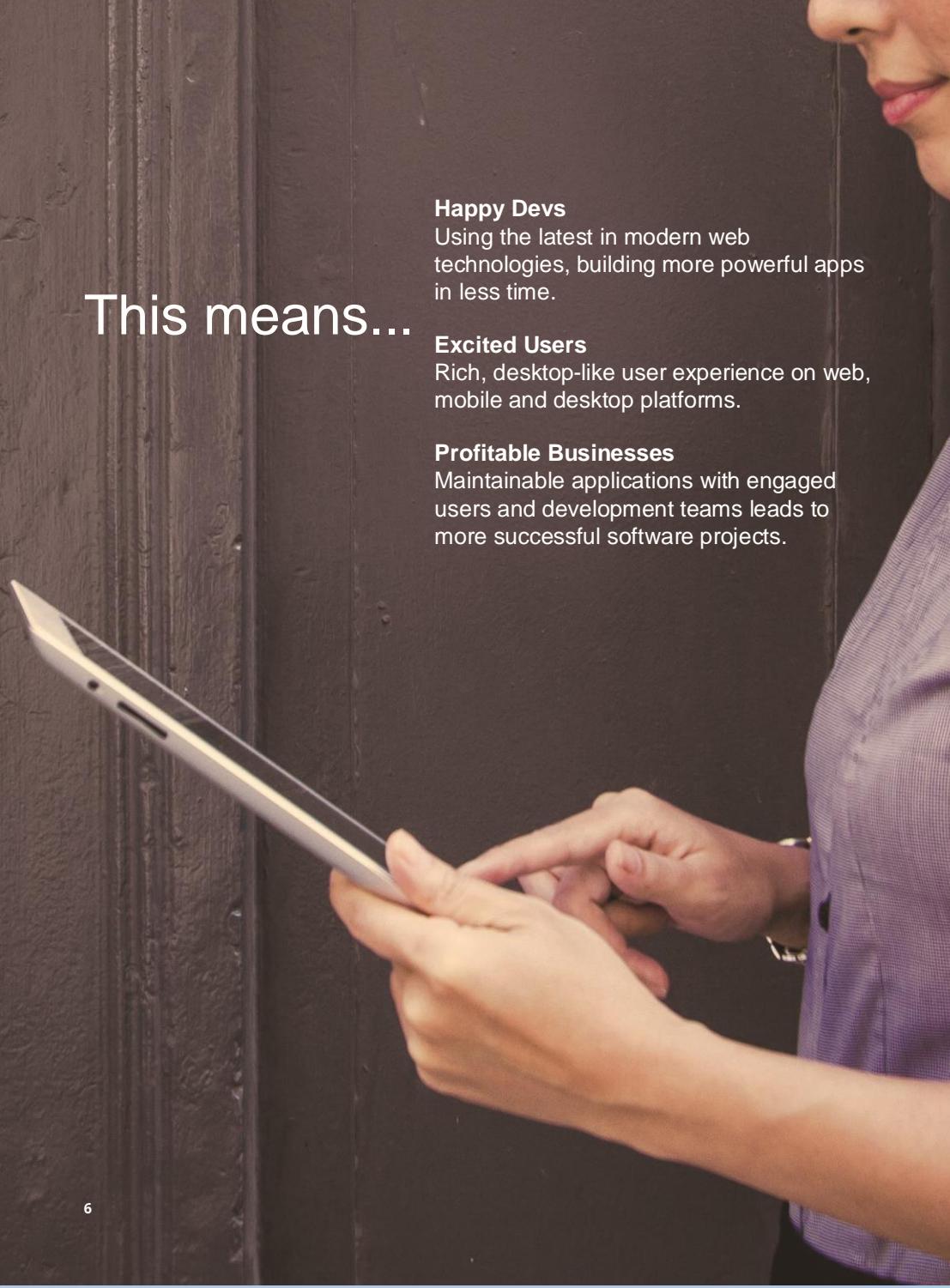
Create high performance, complex choreographies and animation timelines with very little code through Angular's intuitive API.

Accessibility

Create accessible applications with ARIA-enabled components, developer guides, and built-in a11y test infrastructure.

Angular is Enterprise



A photograph of a woman with long dark hair, wearing a purple button-down shirt. She is smiling and holding a silver tablet computer in her hands, looking at the screen. The background is a dark, textured wall.

This means...

Happy Devs

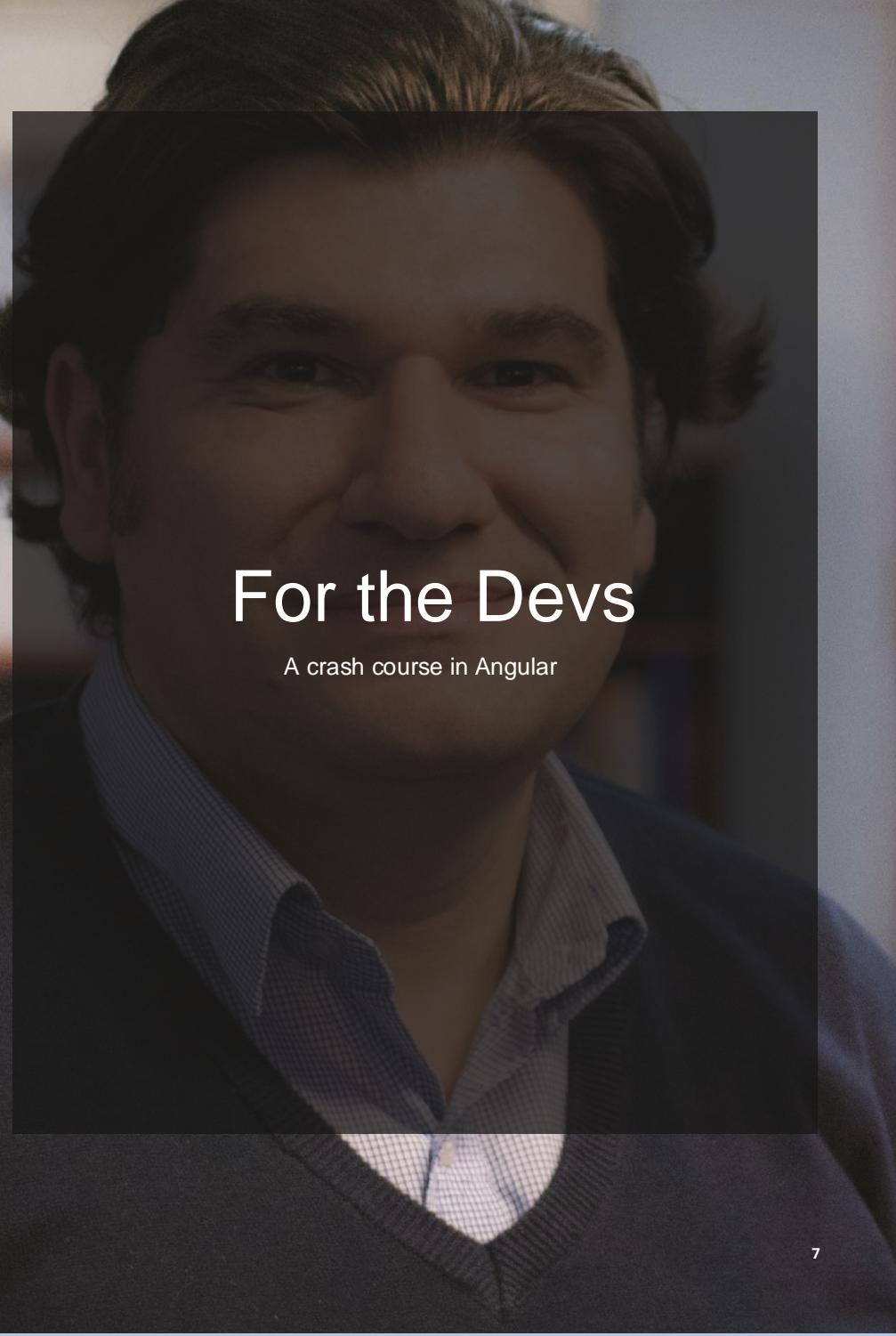
Using the latest in modern web technologies, building more powerful apps in less time.

Excited Users

Rich, desktop-like user experience on web, mobile and desktop platforms.

Profitable Businesses

Maintainable applications with engaged users and development teams leads to more successful software projects.

A close-up photograph of a man with dark hair, wearing a dark blue sweater over a light-colored collared shirt. He is smiling warmly at the camera.

For the Devs

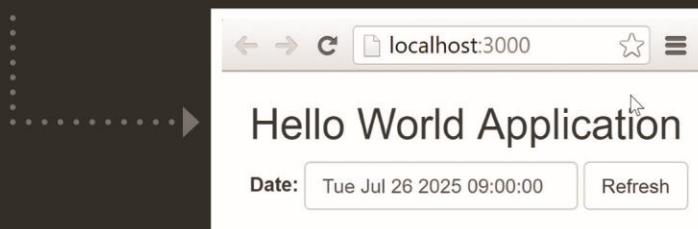
A crash course in Angular

```
hello-world.component.ts
```

```
import { Component } from '@angular/core'; 1

@Component({
  selector: 'hello-world', 2
  template: 3
)
<h2>{{appTitle}}</h2> 4
<label>Date:</label>
<input [(ngModel)]="currentDateTime" type="text"> 5
<button (click)="setCurrentTime()">Refresh</button>
6
7
export class HelloWorldComponent {
  appTitle: string = 'Hello World Application'; 8
  currentDateTime: Date = new Date();

  setCurrentTime() { 9
    this.currentDateTime = new Date();
  }
}
```



'Hello World' Explained

- 1** Import the Angular core file so that our component code can have access to the @Component decorator.
- 2** @Component decorator adds metadata that tells Angular how to create and use this component.
- 3** The selector specifies a simple CSS selector for an HTML element that represents the component.
- 4** Component's companion template, written in an enhanced form of HTML that tells Angular how to render this component's view. Templates can also be stored in external HTML files.
- 5** Double-curly braces, {{ }}, used to bind values onto the page.
- 6** [(ngModel)] two-way data binding syntax to both display a data property and update that property on the component model.
- 7** Event binding syntax consists of a target event within parentheses on the left of an equal sign, and a quoted template statement on the right that often refers to a function on the component class.
- 8** An Angular class responsible for exposing data to a view and handling most of the view's display and user-interaction logic.
- 9** Set public appTitle property as type 'string' and initialize its value as 'Hello World Application'.
- 10** setCurrentTime() is a JavaScript function using the ES6 shorthand syntax.

'Todo' App component tree

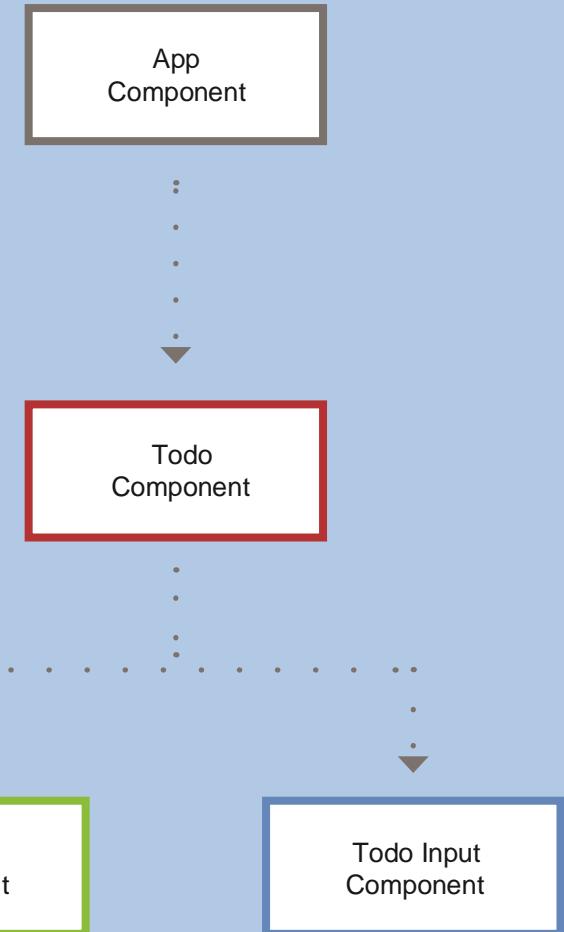
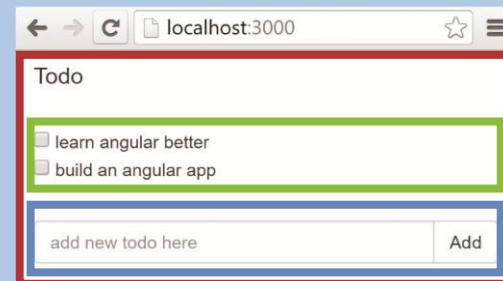
Let's move beyond Hello World and introduce something closer to the real world. We'll begin working on a new Angular component that supports a Todo list.

An Angular application is a tree of components.

The top-level component contains several child components, and each of those in turn is a parent to their child components. The App Component is the parent of the Todo Component, which is the parent of the Todo List Component and the Todo Input Component.

The benefits of the component tree:

- Composability
- Fast change detection strategies
- Component communication between parent and child



todo.component.ts

```

import {Component, OnInit} from '@angular/core';   1
import {Todo} from './todo.model';

@Component({ 2
  selector: 'app-todo',
  template: `

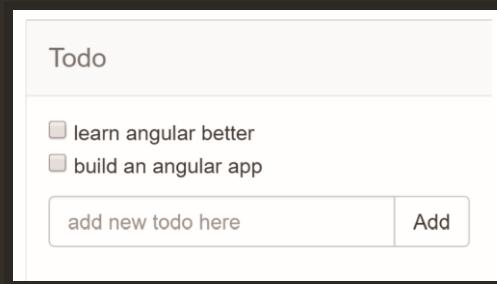
<div>
  <h2>Todo</h2>
  <todo-list [todos]="todos"></todo-list> 3
  <todo-form (newTodoAdded)="addTodo($event)"></todo-form> 4
</div>
` ,
})
export class TodoComponent implements OnInit {
  todos: Todo[]; 5

  constructor(private todoService: TodoService) { }

  ngOnInit() { 6
    this.todos = this.todoService.get()
  }

  addTodo(todo: Todo) { 7
    this.todoService.push(todo);
  }
}

```



'Todo' App Explained

This code sample shows a typical component with injected dependencies and nested components.

- 1 Import Component, OnInit and Todo interface to be available in this file.
- 2 @Component decorator adds metadata that tells Angular how to create and use this component.
- 3 [] square bracket property binding syntax represents an input of a property into a component/directive.
- 4 () parenthesis represents an output of an event from a component/directive.
- 5 Class constructor defining a component's injected dependencies. Angular has its own dependency injection framework. We utilize TypeScript's constructor syntax for declaring parameters and properties simultaneously.
- 6 Components have a lifecycle managed by Angular as it creates, updates and destroys them. Here we use ngOnInit to run state initialization logic.
- 7 addTodo(todo: Todo) is a JavaScript function using the ES6 shorthand syntax.

Module Loader

Angular embraces the new ECMA Script 6 module loading syntax, allowing for better optimized bundles of code to be shipped to the browser.

A module is simply a JavaScript file written with module syntax. Modules export values, which can then be imported by other modules.

A module loader provides the ability to dynamically load modules, and also keeps track of all loaded modules in a module registry.

`math.ts`

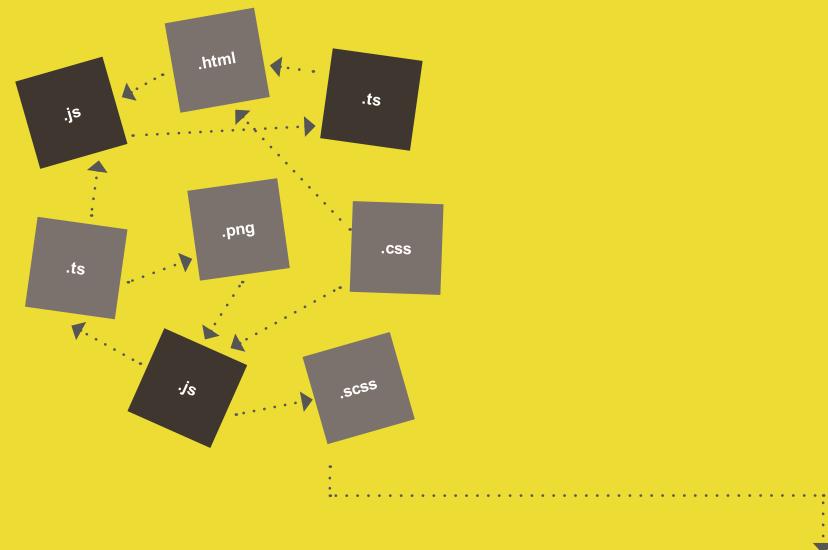
```
export function sum (x, y) {
  return x + y;
}
```

`calculator.ts`

```
import {sum} from './math';

sum(1, 2);
```

ES6 MODULES

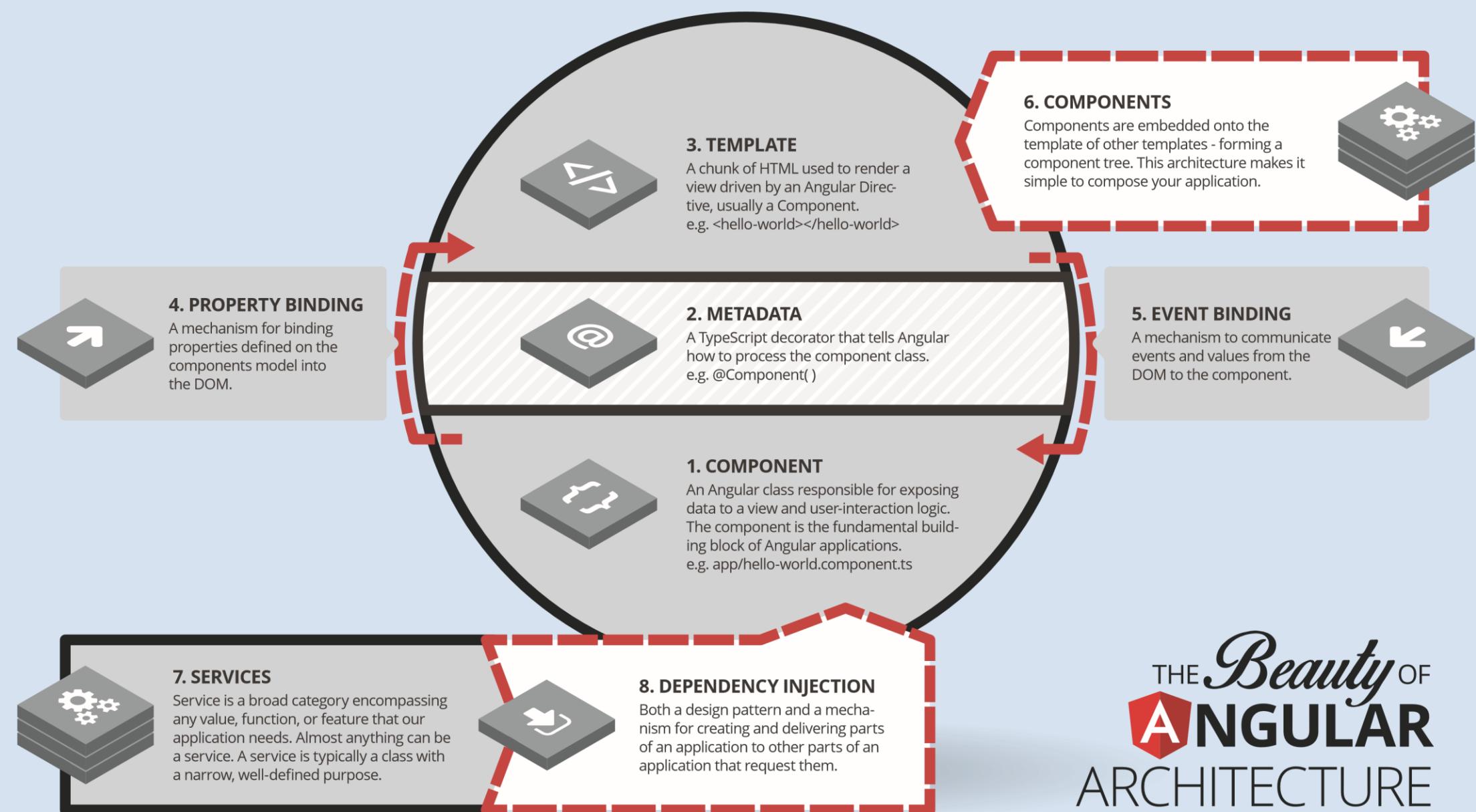


Typically, in production, the module registry would be populated by an initial compiled bundle of modules, which is what module bundlers like webpack and SystemJS/JSPM can do.

Later in the page state, it may become necessary to dynamically load a new module. This module can then share dependencies with the initial page bundle without having to reload any dependencies.

Module Bundler





THE *Beauty* OF
ANGULAR
ARCHITECTURE

TypeScript

TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. Any browser. Any host. Any OS. Open source.

TypeScript is not only the language of choice - although you can use JavaScript or Dart - but most third party libraries, docs, blog posts, and the actual Angular code base is written in TypeScript.

- **Compile Time Checking**

Errors in TypeScript are shown in the IDE as you write your code. The TypeScript compiler will fail with detailed error messages without having to leave your IDE.

- **Great Tooling**

TypeScript's type system allows for advanced autocompletion, navigation, and refactoring. These are almost a requirement for large projects to reduce fear of refactoring a large code base.

- **Easier Code To Read**

With interfaces, strongly typed function parameters and return types, TypeScript code is much more explicit, making it easy to understand.

- **Built In Compiler**

If you use the latest ES6 JavaScript language features you need to compile your JavaScript to ES5 browser compatible code. TypeScript's compiler by default can compile down to ES3, ES5 or ES6.

greeter.ts

```
class Greeter {
  greeting: string;
  constructor(message: string) {
    this.greeting = message;
  }
  greet() {
    return 'Hello, ' + this.greeting;
  }
}

let greeter = new Greeter('world');
```



RxJS

ReactiveX combines the Observer pattern with the Iterator pattern and functional programming with collections to fill the need for an ideal way of managing sequences of events.

RxJS is an essential ingredient of Angular and simplifies asynchronous code

Search...

typeahead.component.ts

```
this.albums$ = this.searchText.valueChanges
  .pipe(
    debounceTime(500),
    distinctUntilChanged(),
    switchMap(searchText => this.albumService.search(searchText))
  )
```

RxJS has over one hundred operators and is why some people think of RxJS as Lodash for events. This four line typeahead example would take many more lines of code, be less expressive and less performant without RxJS.

Operators used:

- **debounce(500)**
Only emit the next search text field value every 500ms.
- **distinctUntilChanged()**
Only emit the next search text field value if it has changed.
- **switchMap()**
Map search text field values into a new observable of http search requests.

The redux pattern is a way to implement a predictable state container for JavaScript apps

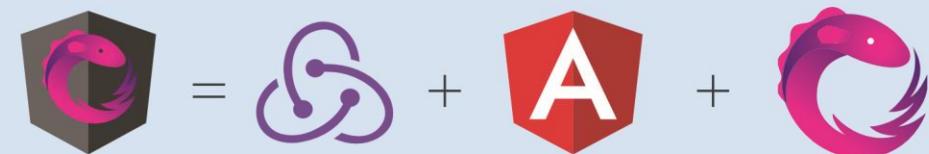
Redux



The redux pattern helps you manage and simplify your application state. Redux (big R) is a library and redux (little r) is a design pattern that is completely framework agnostic.

Three principles of the redux pattern

- The entire state of the application is represented in a single JavaScript object called a store.
- The store is acted upon using special functions called reducers.
- State is immutable and reducers are the only part of the application that can change state.



ngrx builds on the concepts made popular by Redux, by supercharging it with RxJS.



The result is a tool and philosophy that will revolutionize your applications and development experience

Angular CLI



```
> npm install -g @angular/cli
> ng new my-dream-app
> cd my-dream-app
> ng serve
```

ng new

The Angular CLI makes it easy to create an application that already works, right out of the box. It already follows our best practices!

ng generate

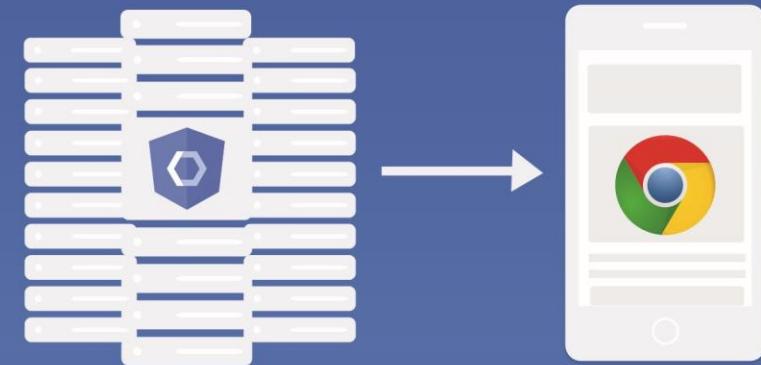
Generate components, routes, services and pipes with a simple command. The CLI will also create simple test shells for all of these.

ng serve

Easily put your application in production

Test, Lint, Format

Make your code really shine. Run your unit tests or your end-to-end tests with the breeze of a command. Execute the official Angular linter and run clang-format.



Angular elements are Angular components packaged as custom elements, a web standard for defining new HTML elements in a framework-agnostic way.

Embedding Angular Components Into Content Sites

If you have a server-side (e.g. ASP.NET) application you want to sprinkle with some Angular magic, Angular Elements are the easiest way to do it.

Embedding Angular Components Into Non-Angular Applications

All frameworks speak “custom elements,” to at least some degree. So, if you want to embed an Angular component into your React, Vue, or Ember app, just wrap it into a custom element.

Angular Elements

**Angular components
packaged as custom
HTML elements**

Progressive Web Apps

Angular CLI PWA toolkit helps you build apps that are:

Progressive

Work for every user, regardless of browser choice because they're built with progressive enhancement as a core tenet.

Responsive

Fit any form factor, desktop, mobile, tablet, or whatever is next.

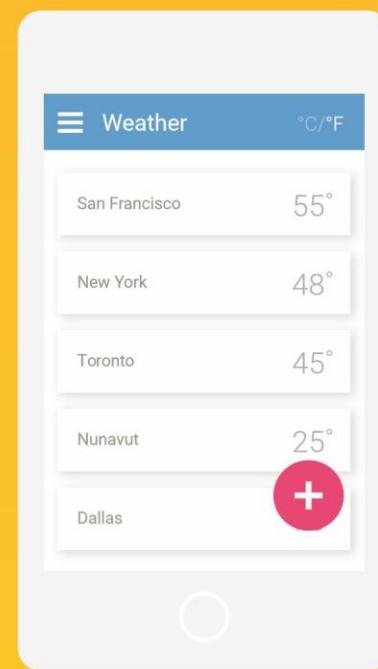
App-like

Use the app-shell model to provide app-style navigations and interactions.

Discoverable

Are identifiable as "applications" thanks to W3C manifests and service worker registration scope allowing search engines to find them.

PWAs are similar to native apps but are deployed and accessible from web servers via URLs, so we don't need to go through app stores.



Angular Material

Material Design components for Angular apps

Sprint from Zero to App

Hit the ground running with comprehensive, modern UI components that work across web, mobile and desktop.

Fast and Consistent

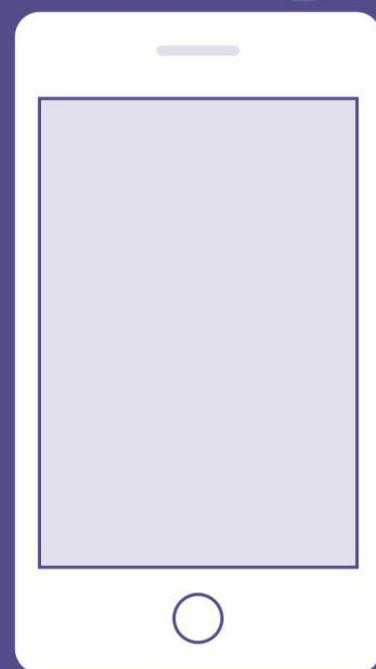
Finely tuned performance, because every millisecond counts. Fully tested across IE11 and current versions of Chrome, Edge, Firefox and Safari.

Versatile

Themable, for when you need to stay on-brand or just have a favorite color. Accessible and internationalized so that all users are welcome.

Optimized for Angular

Built by the Angular team to integrate seamlessly with Angular.



Bright future

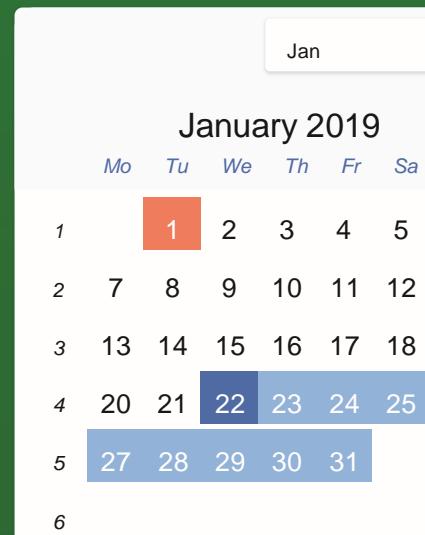
New features keep coming

- Faster development speed via build caching (since Angular 13)
- Nullish coalescing and optional chaining – cleaner code for nullable objects (since Angular 12)
- Better Angular Language Service for VS Code users (since Angular 11)

Ng-bootstrap Now with Angular, we can use the 'ng add' command to do most of the work for us.

Run the following commands to install Twitter Bootstrap for Angular into your project.

```
ng add @ng-bootstrap/ng-bootstrap
```



NSwag provides tools to generate Swagger specifications from existing ASP.NET Web API controllers and client code from these Swagger specifications.

- Generate Angular clients/proxies from your OpenAPI spec
- Everything can be automated via CLI (distributed via NuGet tool or build target; or NPM)
- CLI configured via JSON file or NSwagStudio Windows UI



Learn Angular with the best consultants

DEV SUPERPOWERS[®]

• TOUR •

Learn more in a day than you could in a month

Come watch our best devs build an app live using the latest technologies

<https://firebootcamp.com/superpower-tours/>



Angular is HTML designed for building enterprise web apps. You've heard it all already: cross-platform, lightweight, testable, open-source MVC JavaScript framework.

This ain't the .NET you know. Rewritten to be cross platform for Linux and Mac users, .NET (previously known as .NET Core) boasts better performance and testability – it's even open-source!

Everything's moving to the Cloud, baby! Learn how to build web applications that are going to need less maintenance and scale to millions of users.

The explosive growth of web frameworks and the demands of users have changed the approach to building enterprise applications. Getting started can be a daunting prospect. Let's change that now.

HANDS-ON WORKSHOP

Our Hands-On Workshops are 2 day events, where you build the app! You will be coached by an SSW FireBootCamp mentor and walk away with the source code, the course material and the new confidence.

firebootcamp.com/hands-on-workshops/



SSW is the consulting company behind FireBootCamp. SSW has over 25 years of experience training developers and developing awesome Microsoft solutions using Angular, Azure, TFS, SharePoint, .NET, Dynamics CRM and SQL Server.

In 1999, we were first recognized as a Microsoft Gold Partner. Today SSW has competencies in a variety of areas, earning a gold in Application Development, Application Lifecycle Management, Collaboration and Content, Digital Advertising, and Management and Virtualisation.

This book was inspired by Ben Cull's '**The Best (and worst) JavaScript Frameworks in Under 2 Minutes!**'



Writers Brendan Richards · Jean Thiron · Liam Elliot · Duncan Hunter · Adam Stephensen · **Editors** Adam Cogan · Marlon Marescia · **Designer** Rebecca Liu ·

Bibliography Angular.io · TypeScriptLang.org · Redux.js.org · SSW Rules to Better AngularJS · SSW Rules to Better Angular



<🔥>
SSWFireBootCamp

www.firebootcamp.com

