ANGULAR WORKSHOP

WINNIPEG / Prairie Dev Con 2017

Laurent Duveau

Updated: 2017-06-03

Who am I ?

Laurent Duveau

- I live in Montreal, Canada
- Founder of the Angular Academy
- 2-day Angular Classroom Training
- 64 classes in the last 18 months
- Montreal, Boston, Quebec, Toronto, Vancouver, Ottawa, Calgary, London, Copenhagen…
- @LaurentDuveau

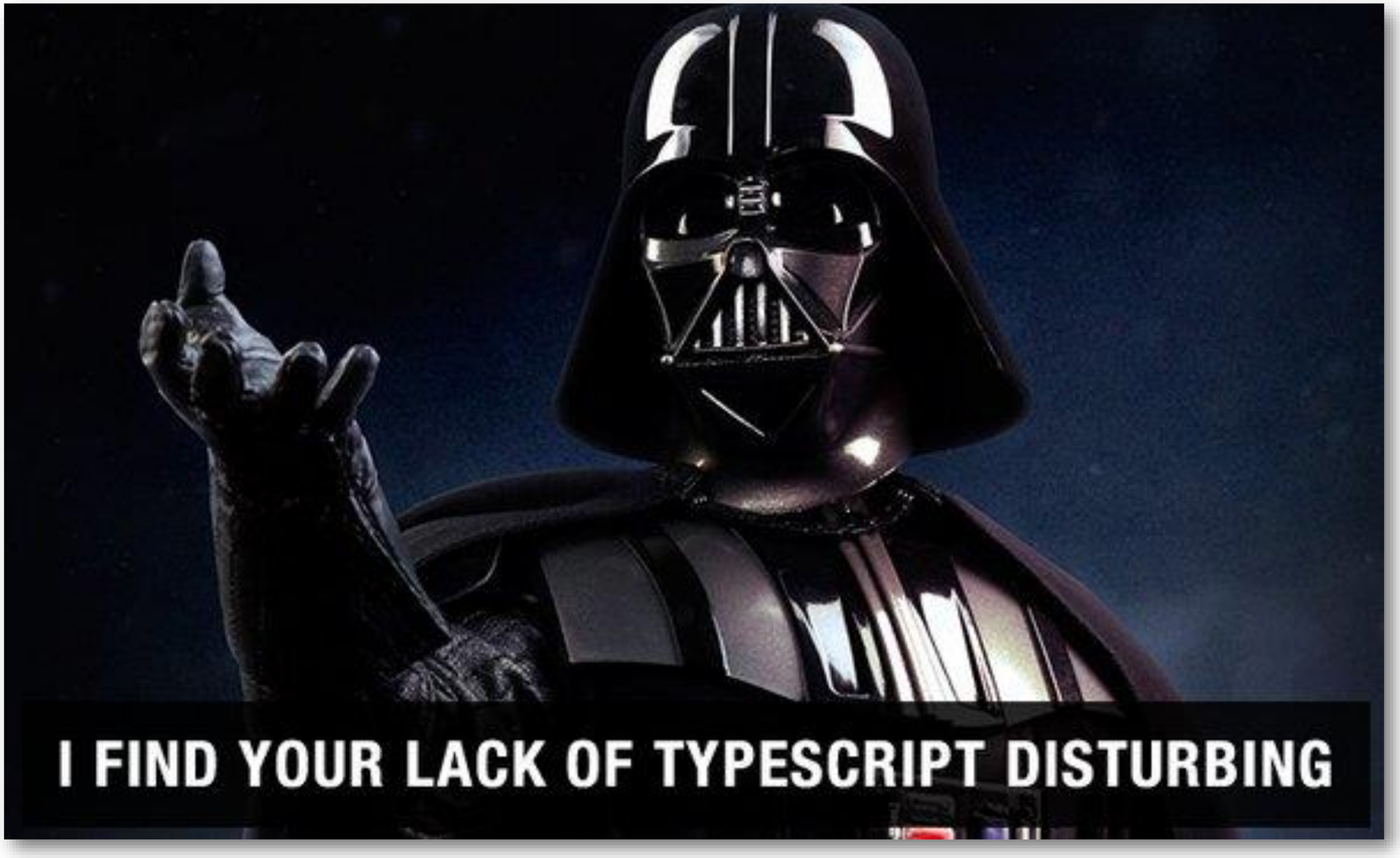
Logistics

- 8:30am - 5:00pm
- Lunch 1 hour
- Morning and afternoon: short 10min. breaks
- WiFi **??**
  - Password: **??**

Agenda

- TypeScript fundamentals
- Setup and Tooling (CLI)
- Introduction to Angular
- Components
- Templating syntax
- Dependency Injection
- Services
- Intro to Reactive Programming with RxJS
- Async Requests to a REST API (HTTP)
- ngModules, AoT Compilation
- Release into production
- *Navigation with the Router and Lazy-loading (if time permits)*

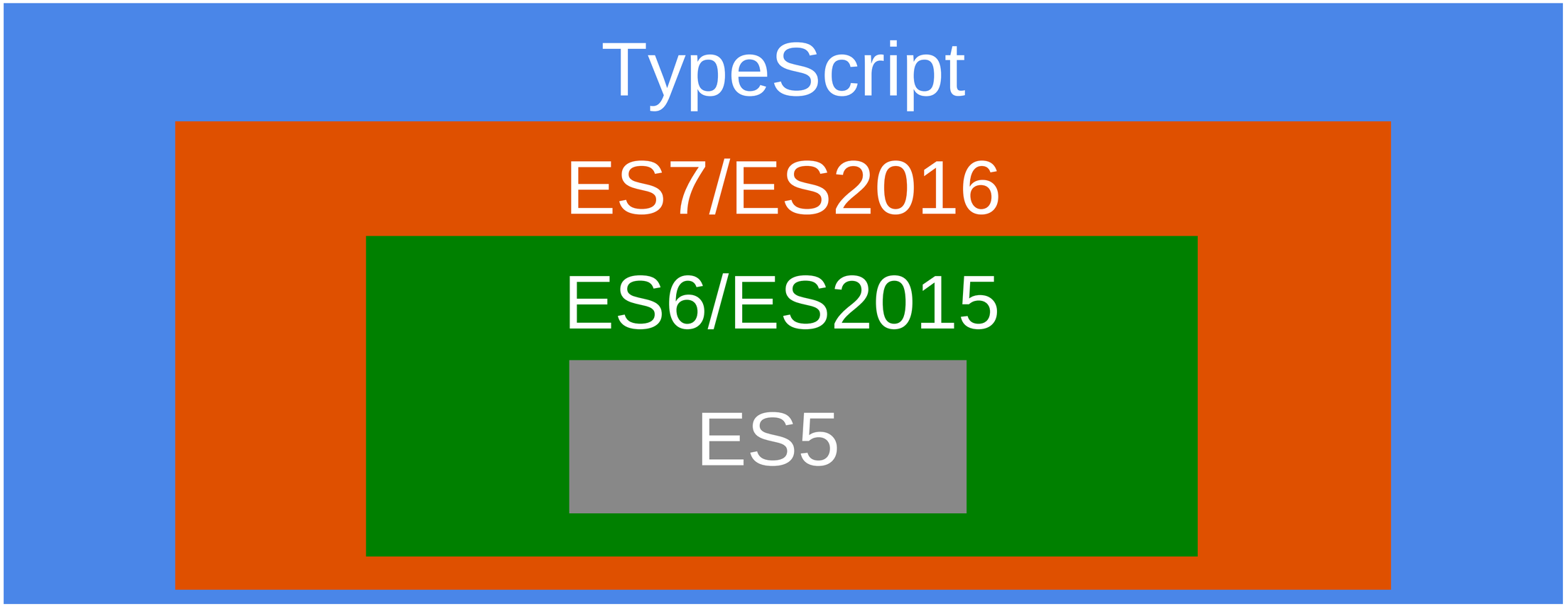
TypeScript


I FIND YOUR LACK OF TYPESCRIPT DISTURBING

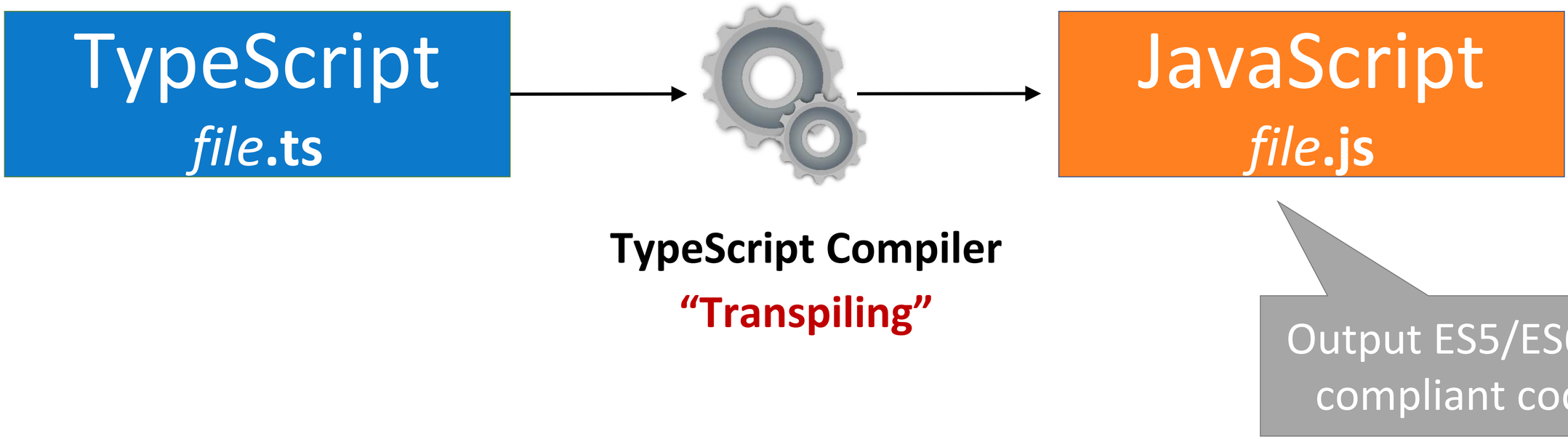
What is TypeScript?

- TypeScript is a **typed** superset of JavaScript that **compiles** to plain JavaScript
- Any browser. Any host. Any OS
- It is a Microsoft Technology, **Open Source**
- You can combine Javascript with Typescript

TypeScript is a Superset of JavaScript



How Does TypeScript Work?



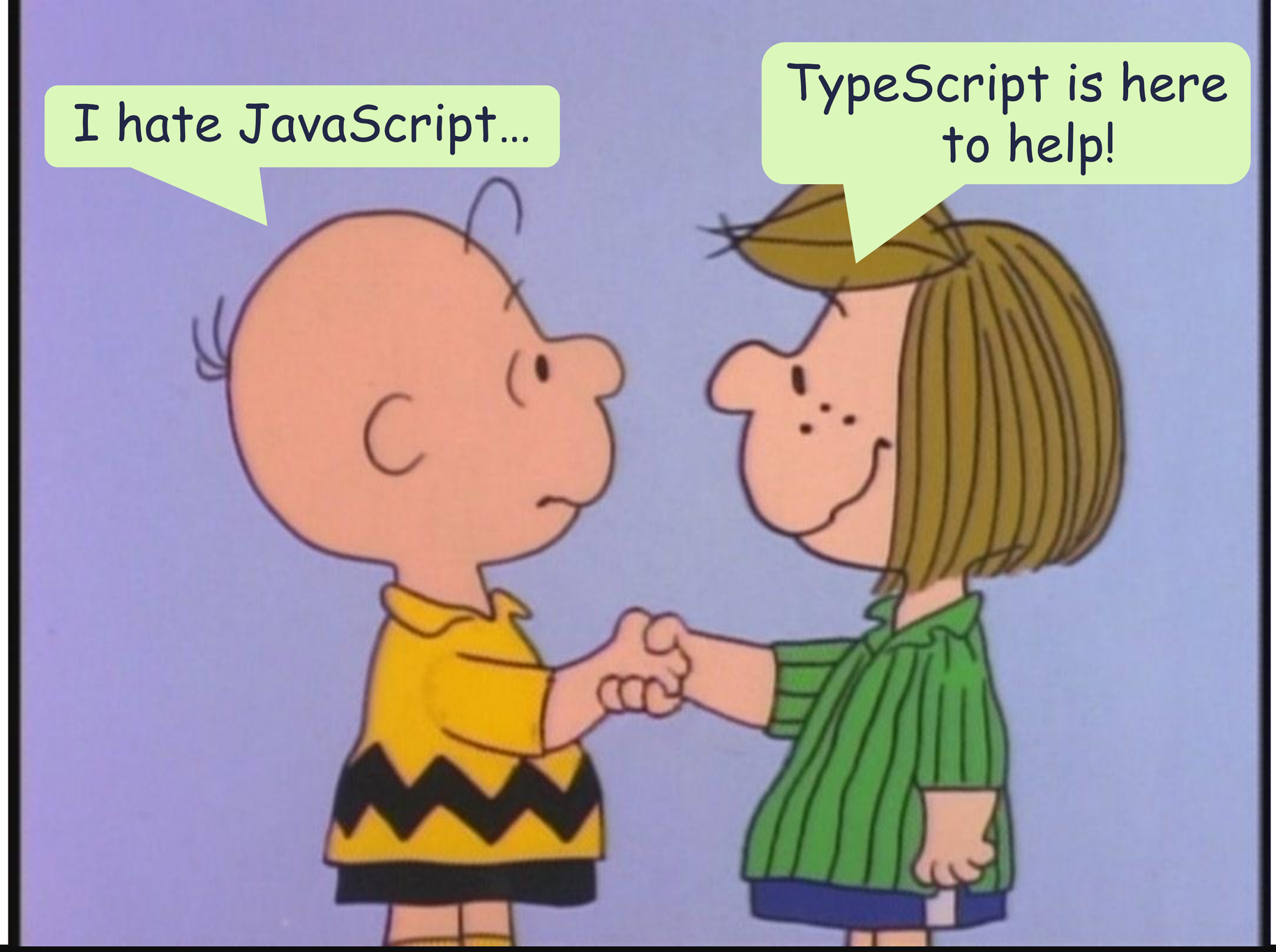
tsconfig.json

```
{
  "compilerOptions": {
    "target": "es5",
    "module": "system",
    "moduleResolution": "node",
    "sourceMap": true,
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "removeComments": false,
    "noImplicitAny": true,
    "suppressImplicitAnyIndexErrors": true,
    "allowJs": true
  }
}
```

Target to Transpile to

Map files for live debugging!

Any Editor - Any OS


I hate JavaScript…
TypeScript is here to help!

Type annotations

```
let height:number = 6;
let isDone:boolean = true;
let name:string = "Angular Academy";
let list:number[] = [1, 2, 3];

function add(x: number, y: number): number {
    return x + y;
}

let res = add(18, "5"); // TypeScript error!
```

Generics, multi types and any

```
let list: Array<number>;
list = [17, 99, 42]; // OK

let cpt: number|string;
cpt = 123; // OK
cpt = "123"; // OK

let z: any;
z = GetJsonObject();
```

Interfaces

```
interface IPerson {
    firstName : string;
    lastName?: string; // ? means optional
    sayHi:() => void;
}
function receivePerson(person:IPerson) {
    person.sayHi();
}
```

DEMONSTRATION / CODE ALONG

"Angular technically doesn't require TypeScript kind of like technically a car doesn't require brakes." *– Joe Eames*

TypeScript sessions

- **An Introduction to TypeScript**

Tuesday at 11:00 AM - Jason Bock

- **Doing Cool Things in a Pure TypeScript Project**

Tuesday at 2:15 PM - Alex Trauzzi

- **TypeScript: Angular’s Secret Weapon**

Wednesday at 8:30 AM - Laurent Duveau

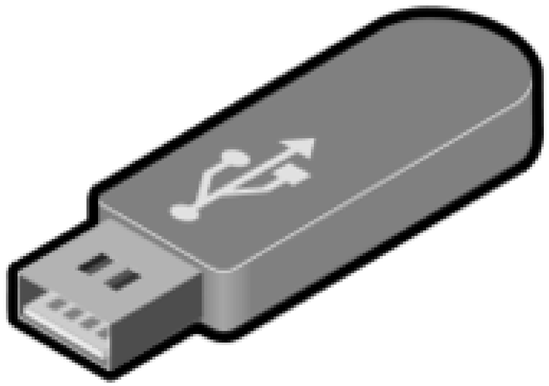
Getting Started

(Setup, configuration and tooling)

Shared files

Download, or get a USB key!

http://angular.ac/ng-workshop

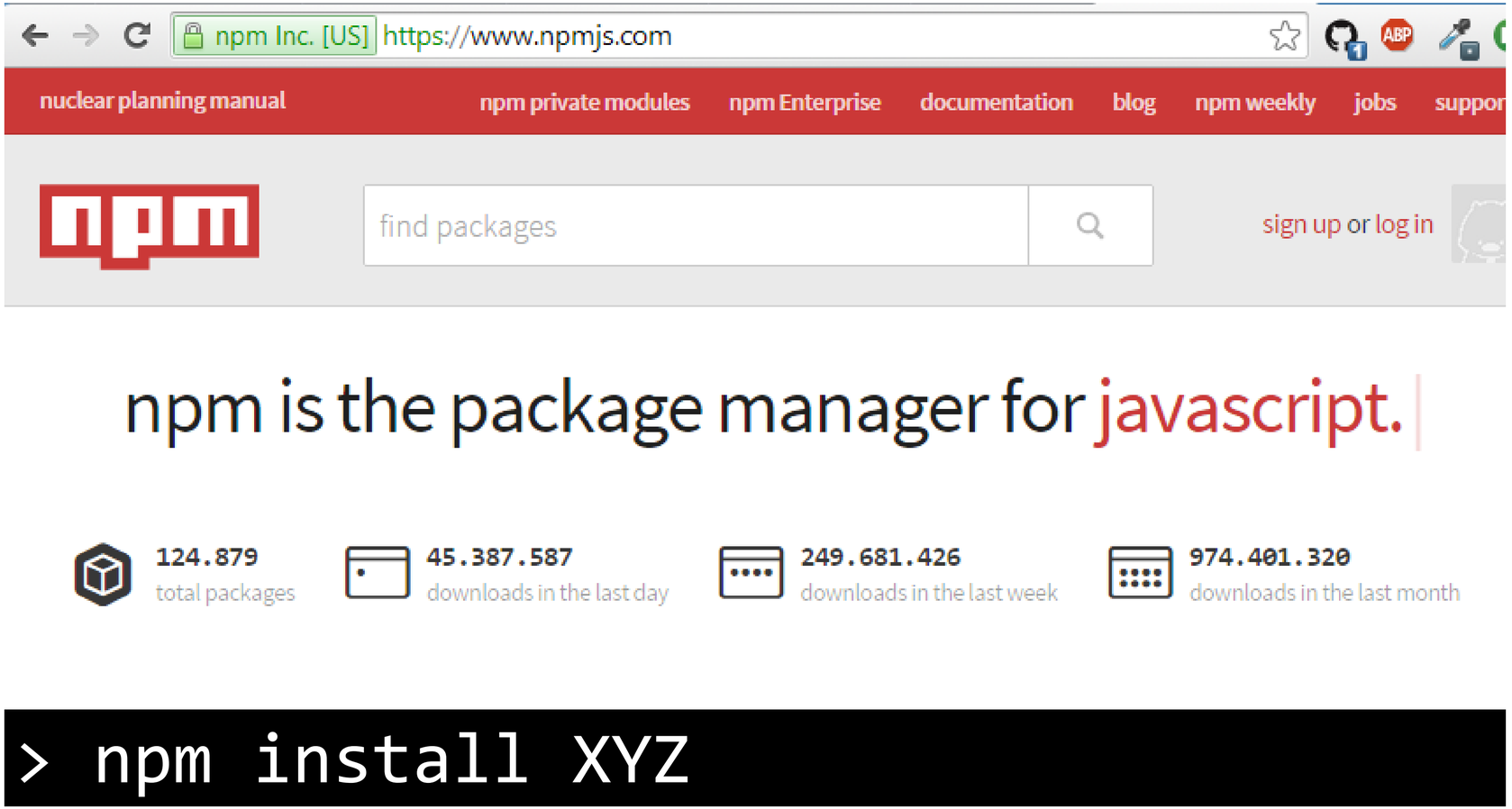


- labs.zip ---------- For the exercises!
- angular.pdf -------- This presentation
- setup.pdf ---------- Setup procedure
- steps.zip ---------- Exercises solutions

Where is my installer ??

No installers…

Getting familiar with npm



Typical environment setup…

- Install NodeJs
- package.json
- tsconfig.json
- typings.json

- Setup grunt/gulp or WebPack
- Setup module loader
- Setup unit testing
- Setup build release

```
> npm install angular typescript typings
tslint karma jasmine ……
```

Typical environment setup…

Angular CLI! (Command Line Interface)

- Getting started faster!

```
> npm install -g @angular/cli

> ng new DemoApp

> ng serve
```

+ generate components, services, routes, …

cli.angular.io

The Angular CLI makes it easy to create an app that works and follows best practices, right out of the box.

```
> ng -v
```

```
> ng serve -o
```

DEMONSTRATION / CODE ALONG

IDEs and code editors

- Sublime Text
- Atom
- Brackets
- WebStorm
- Visual Studio
- …

Visual Studio Code

There's a new kid in town!

- **VS Code**… a code editor, not an IDE
- Free, Open Source
- Windows, Mac, Linux!
- HTML5, JavaScript, CSS, LESS with NodeJs or ASP.NET 5
- Rich code editor from VS in a fast and lightweight tool
- Debug, deploy
- Git integration
- Plenty of extensions

code.visualstudio.com

DEMONSTRATION / CODE ALONG

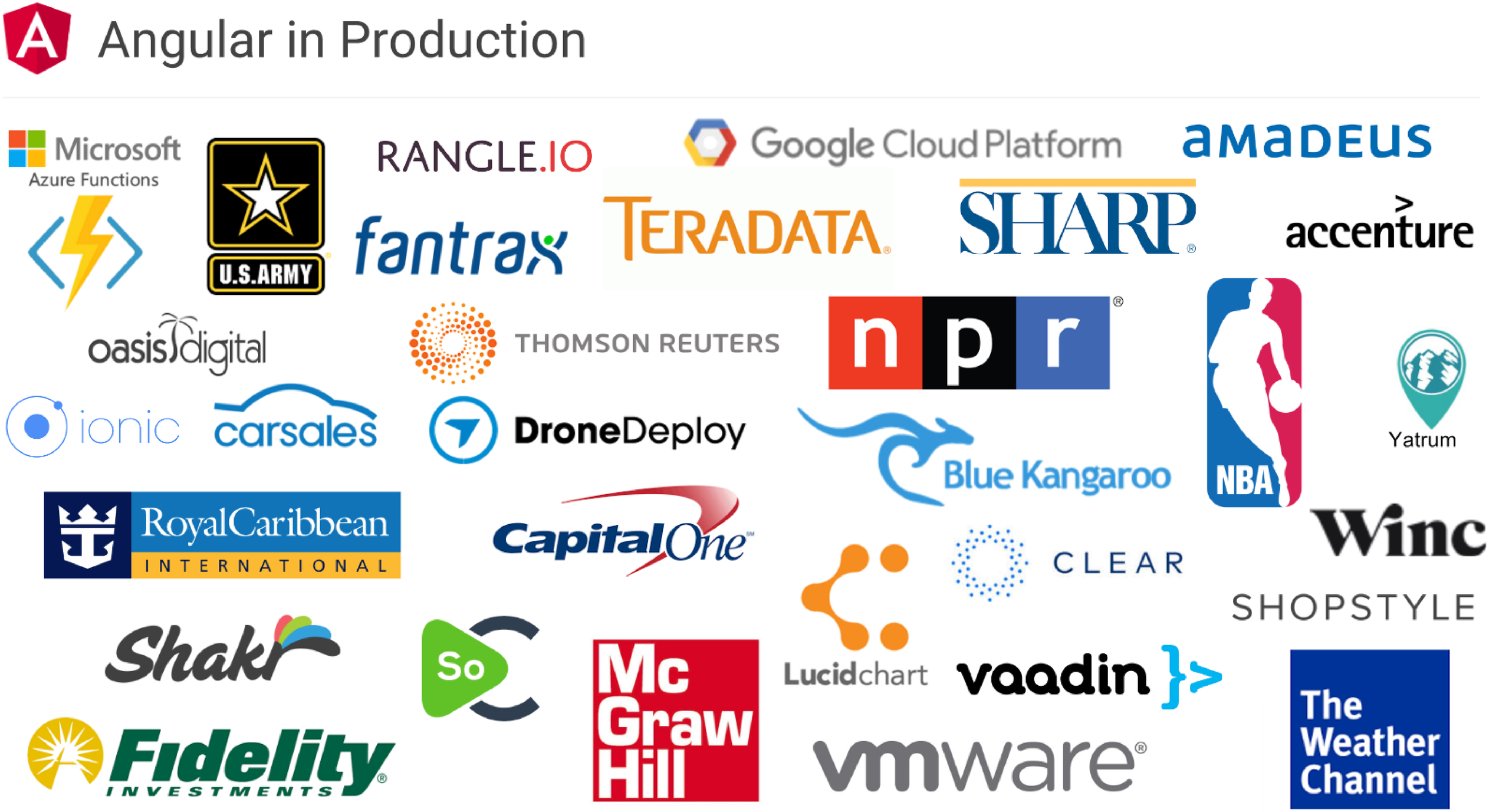
Introduction to Angular

Angular ?

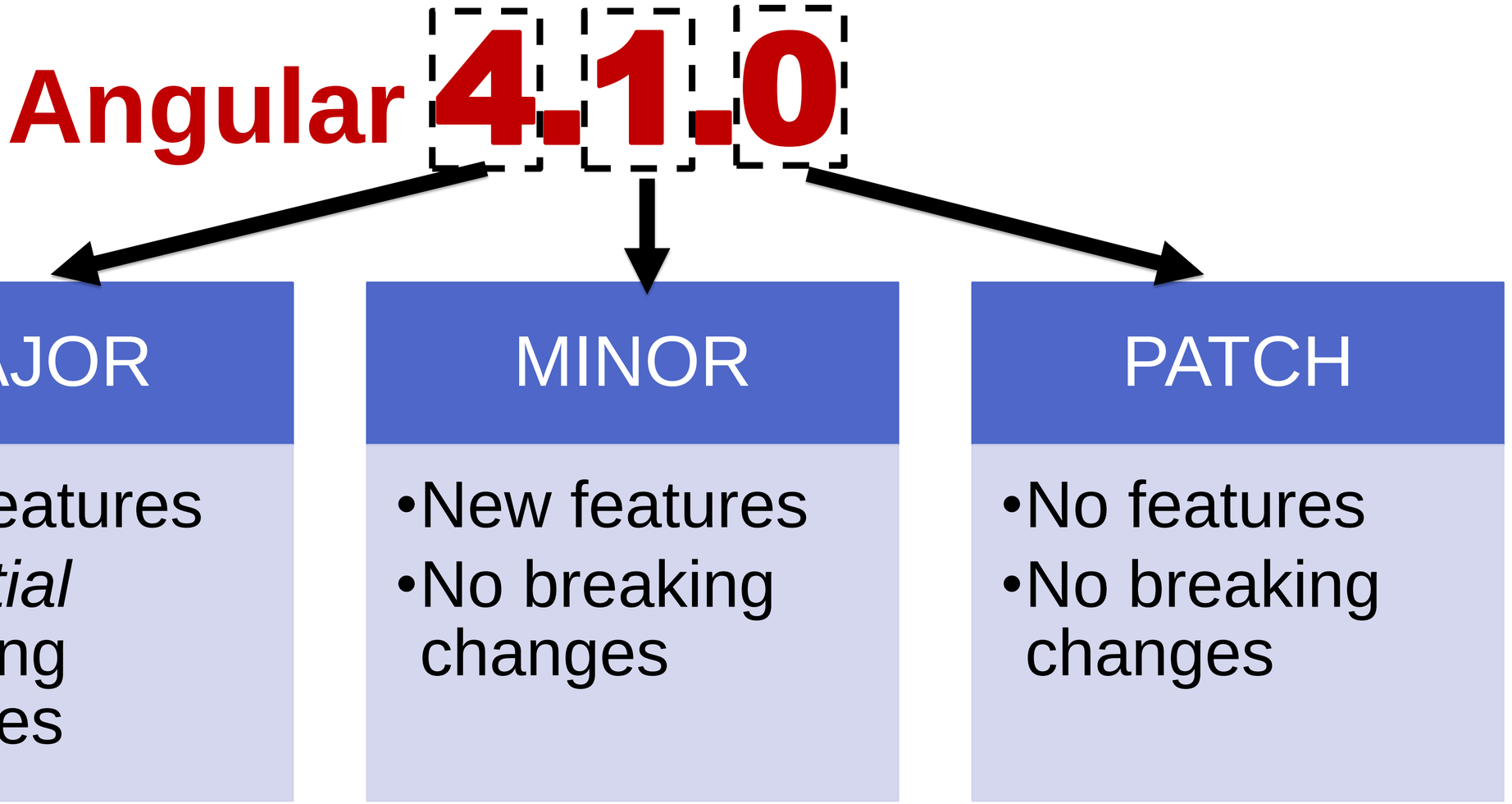
- JavaScript Framework especially suitable for single-page modern web applications (Single Page Application, or SPA)
- Compatible with IE 9+ and others modern browsers
- Open Source, MIT license
- v2 released in September 2016
- v4 released in March 2017

www.angular.io

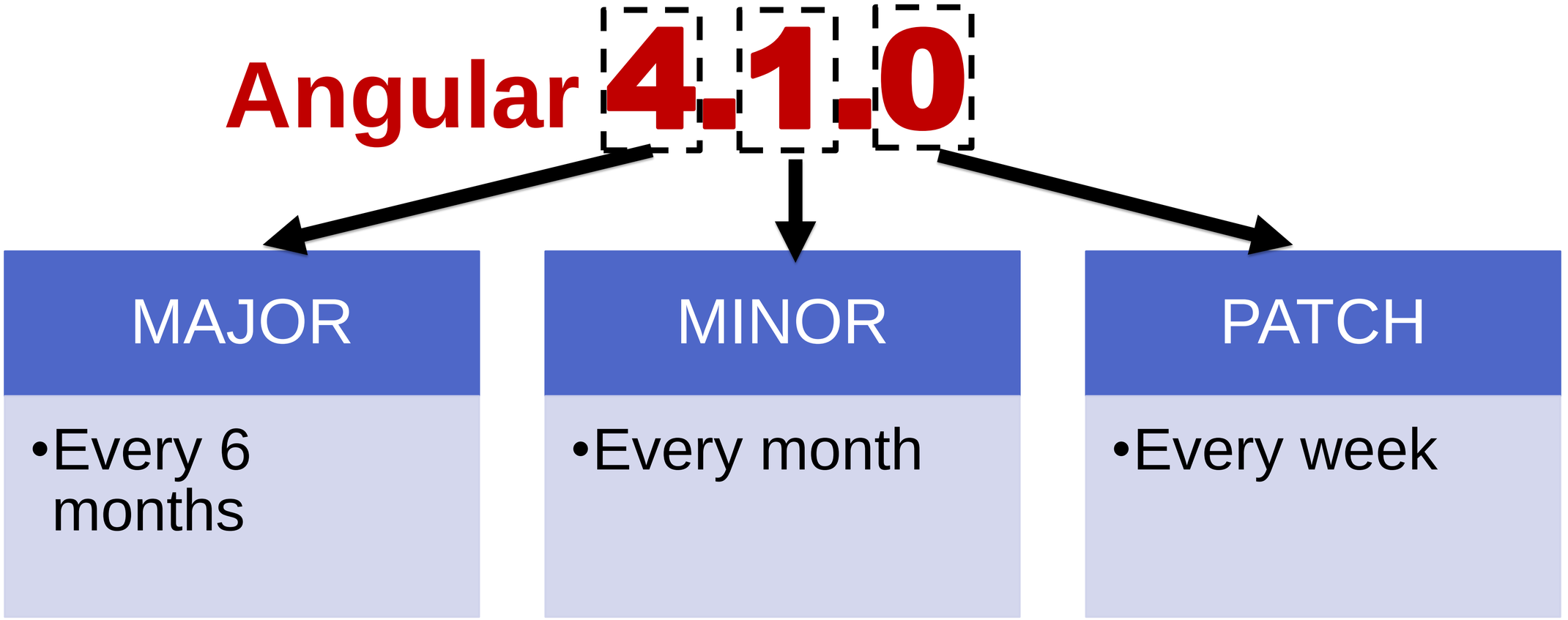
Angular in Production



Semantic Versioning (SemVer)



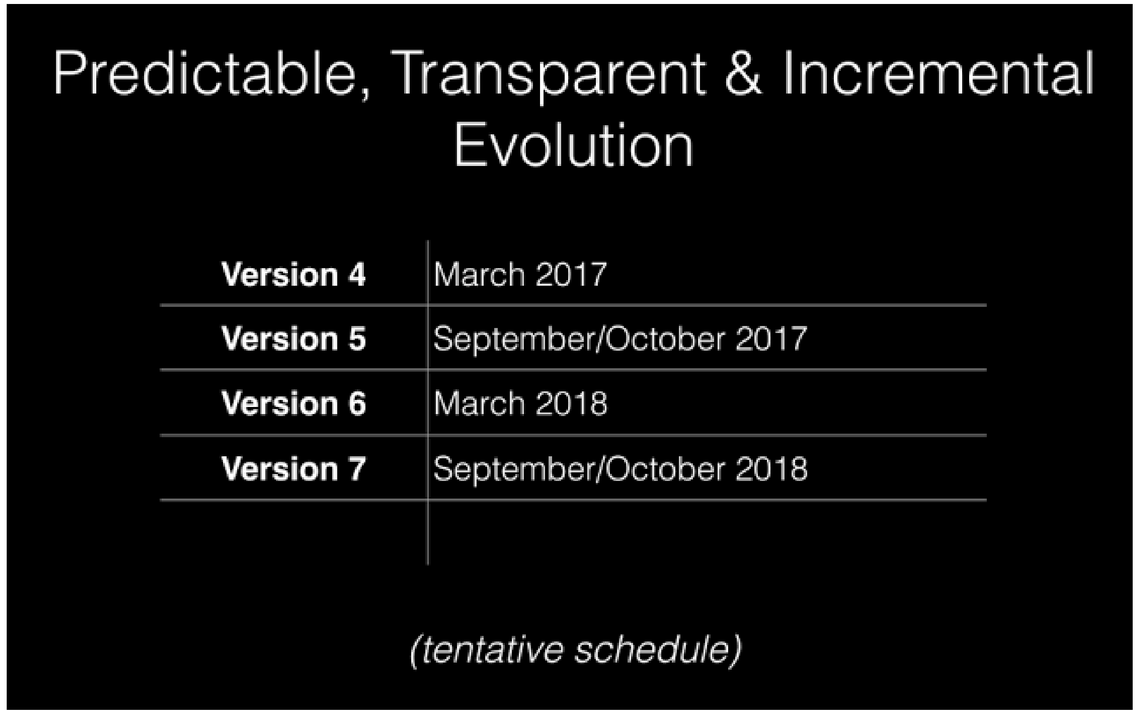
Semantic Versioning (SemVer)



Angular future…

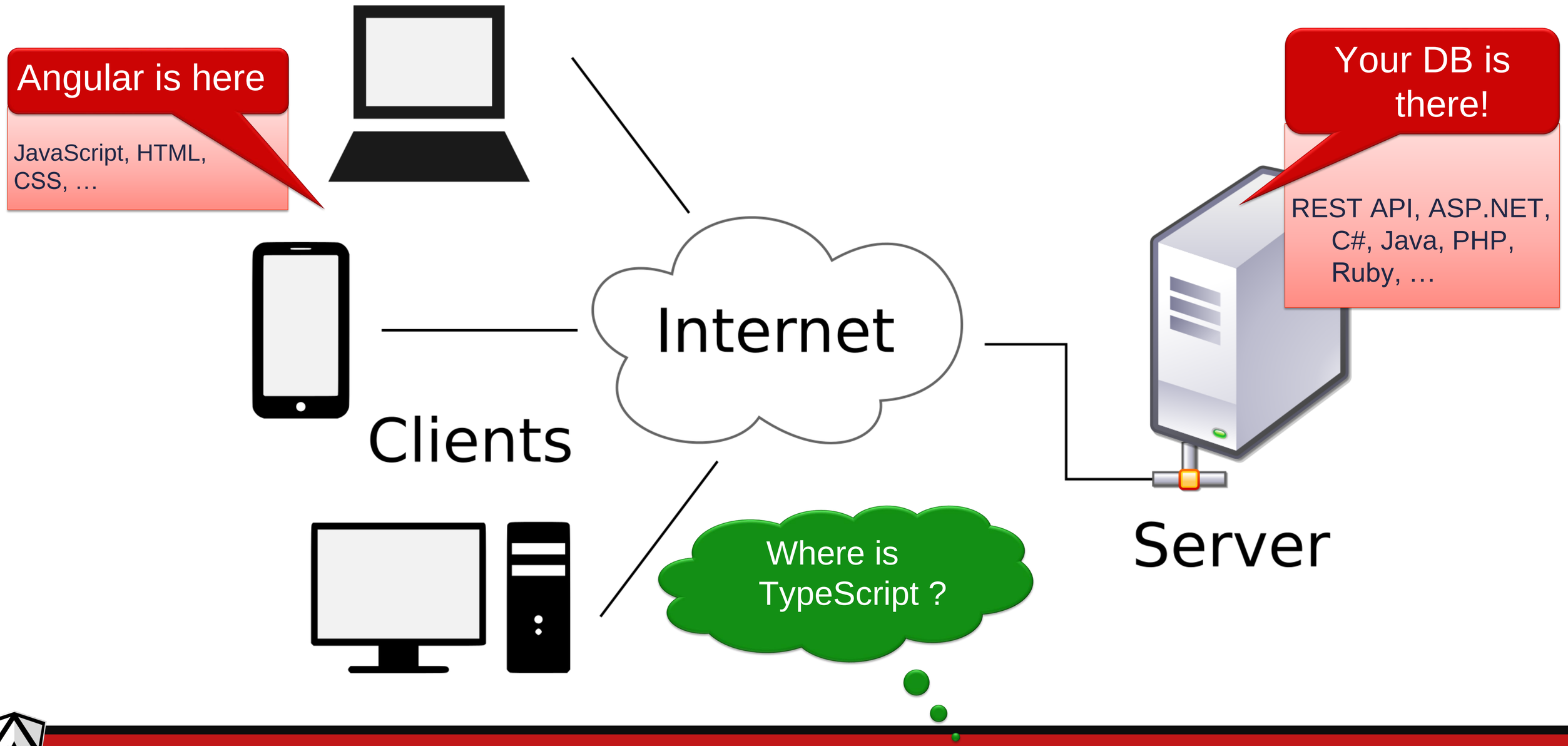
Angular 5 scheduled for Sept. 2017!

- http://angular.ac/justAngular

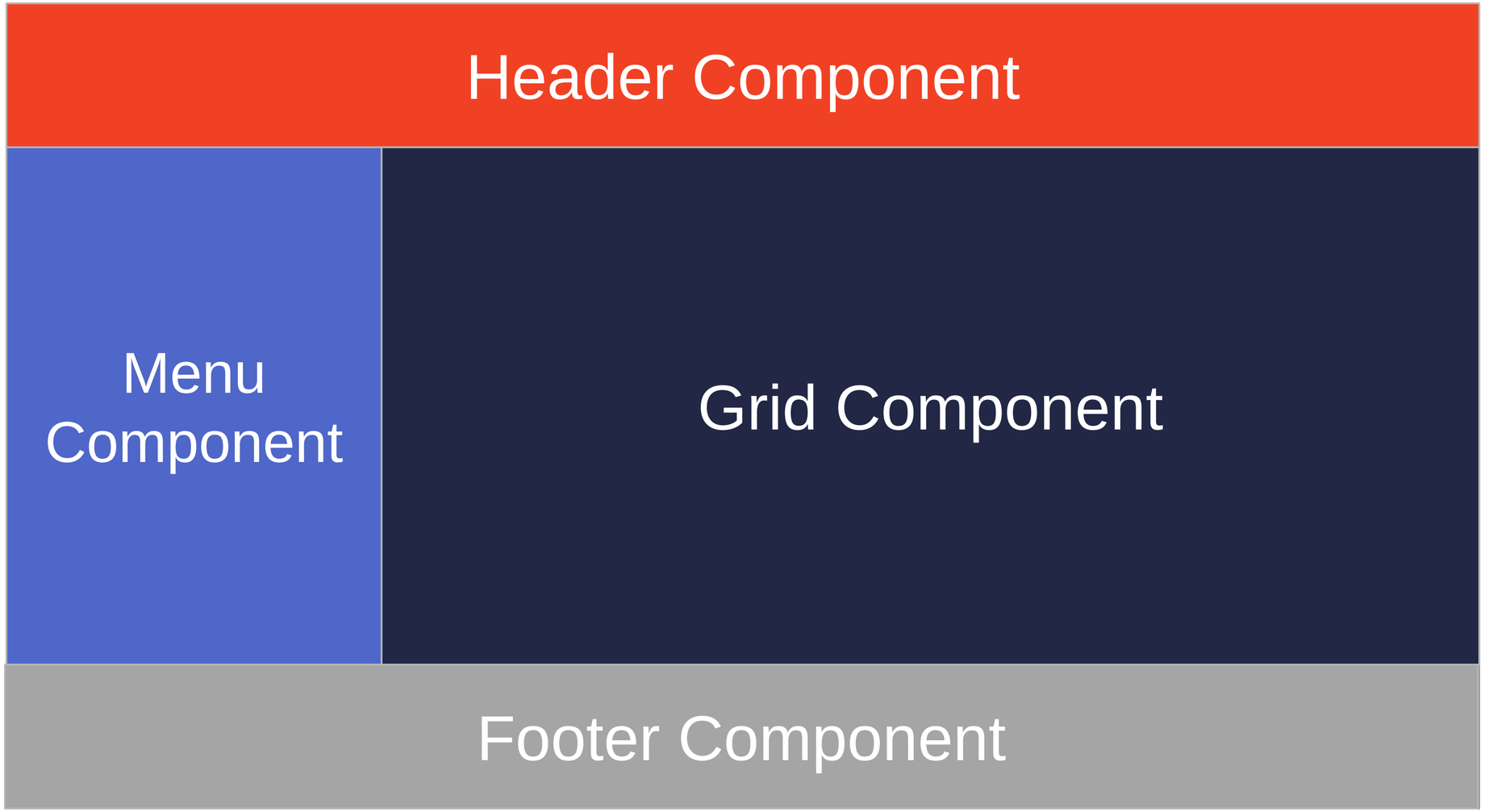


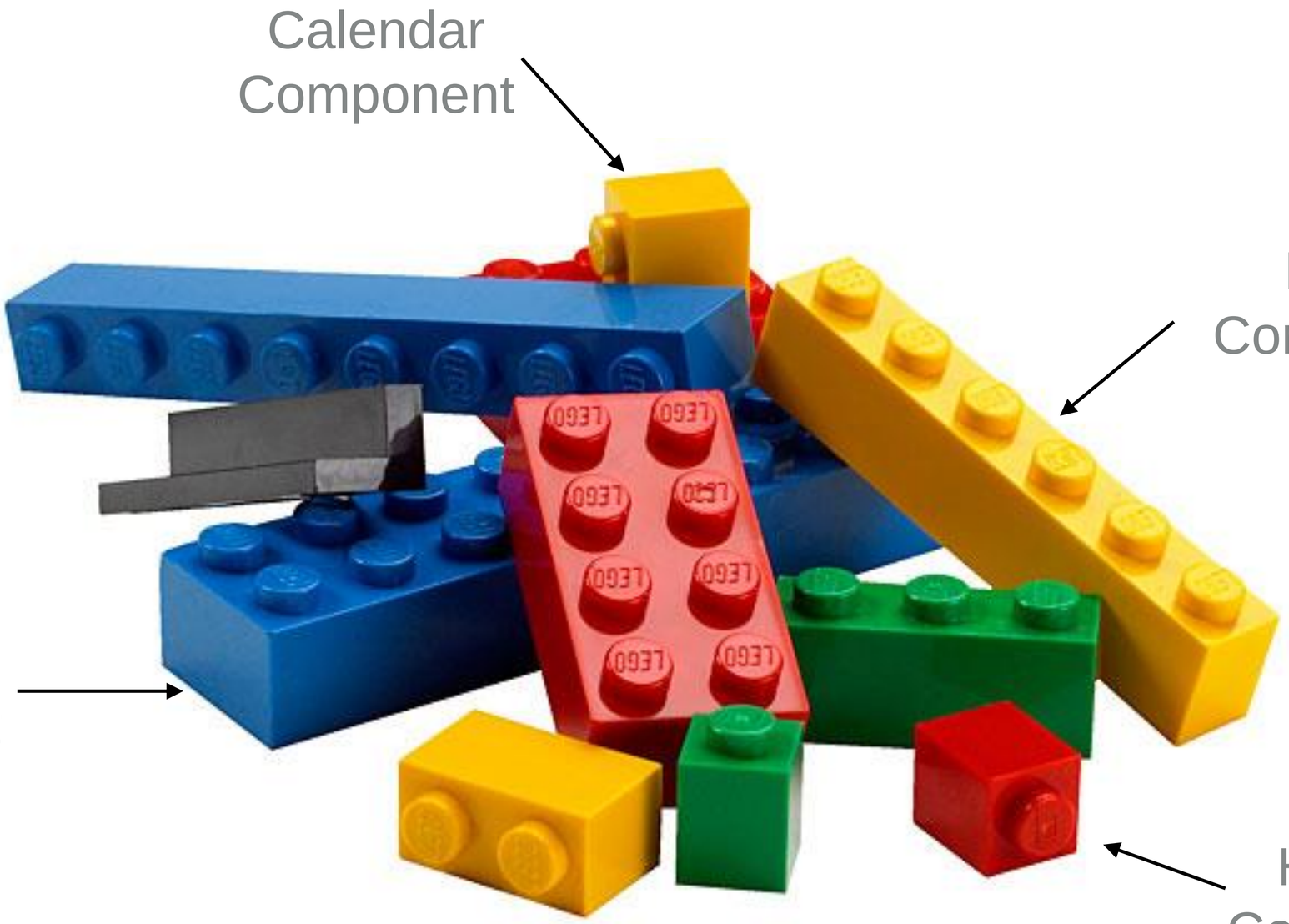
Long Term Support (LTS)

- Angular 4 introduces an LTS
- Critical Bugfixes and Security Patches through October 2018
- Good for stability in enterprises


Angular is here
JavaScript, HTML, CSS, …
Your DB is there!
REST API, ASP.NET, C#, Java, PHP, Ruby, …
Internet
Clients
Server
Where is TypeScript ?

Building SPAs with Components




Calendar
Component
Menu
Component
Grid
Component
Header
Component

Bootstrap to load the app

index.html

```
<html>
<body>

  <app-component></app-component>

  <script src="bundle.js"></script>
  <script src="..."></script>

</body>
</html>
```

Bootstrap

2

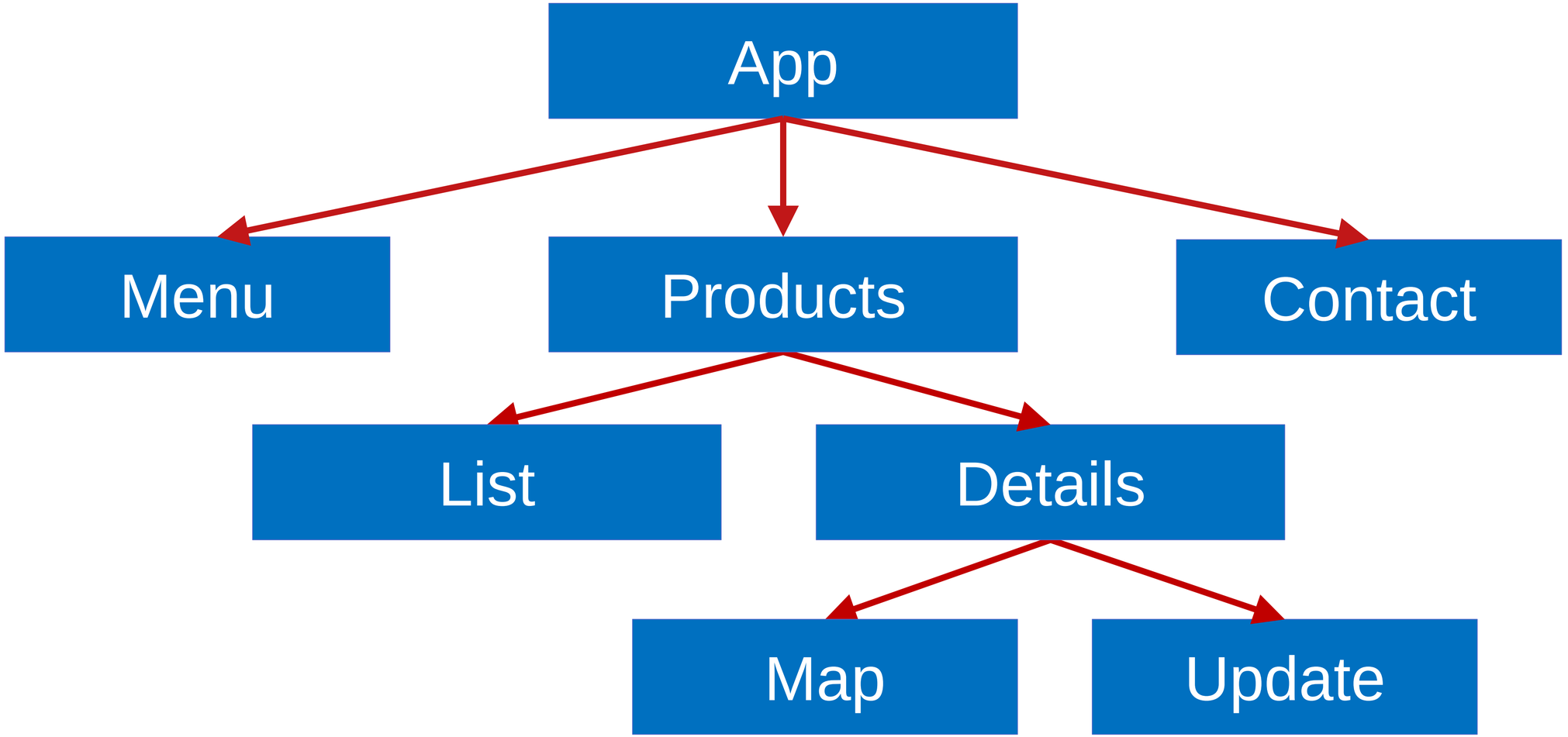
```
@Component({
selector: 'app-component'
})
class AppComponent {

}
```

DEMONSTRATION / CODE ALONG

Components

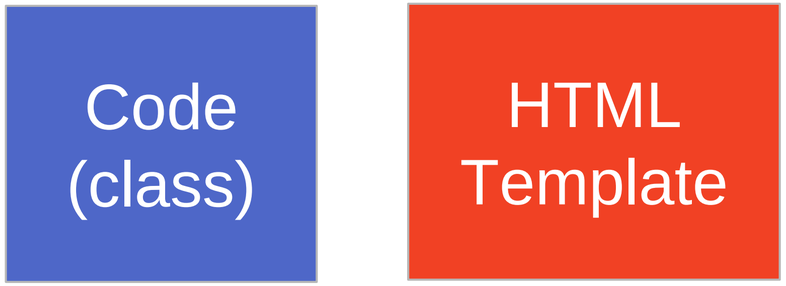
Your app is a tree of components



What is a Component?

- A component is a reusable object

- Made up of:



- Has a "selector": <products-list></products-list>

What's in a Component Class?

imports

Decorators

class

```
import { Component } from '@angular/core';
import { DataService } from './data.service';


@Component({
  selector: 'product-details',
  templateUrl: 'product-details.component.html'
})

export class ProductDetailsComponent {

}
```

ES6 Modules and import/export syntax

A file is an ES6 module

data.service.ts

```
export class DataService {

    ...

}
```

customers.component.ts

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

import { DataService } from '../shared/services/data.service';

...
```

Imported module is relative to current file.

Declarative Templating Syntax

Components and Templates

- Components rely on **decorators** to define metadata, including the path to its HTML template

```
@Component({
  selector: 'customer-info',
  templateUrl: 'app/customers/customers.component.html',
})
export class CustomersComponent {
  constructor(private dataService: DataService) { }
}
```

Rendering

{{ expression }}

```
<div>
  {{ product.name }}
</div>
```

Property Binding

[property]

```
<img [src]="imageUrl" />
<div [hidden]="isHidden"></div>
<div [sty]  idth.px]="mySize"></div>
<my-menu [items]="mainMenu"></my-menu>
```

Bind to any
DOM property

Handling Events

(event)

```
<button (click)="sort($event)">Total</button>

<select (change)="sortProperty('price')">
```

Two-Way Binding

[(ngModel)]

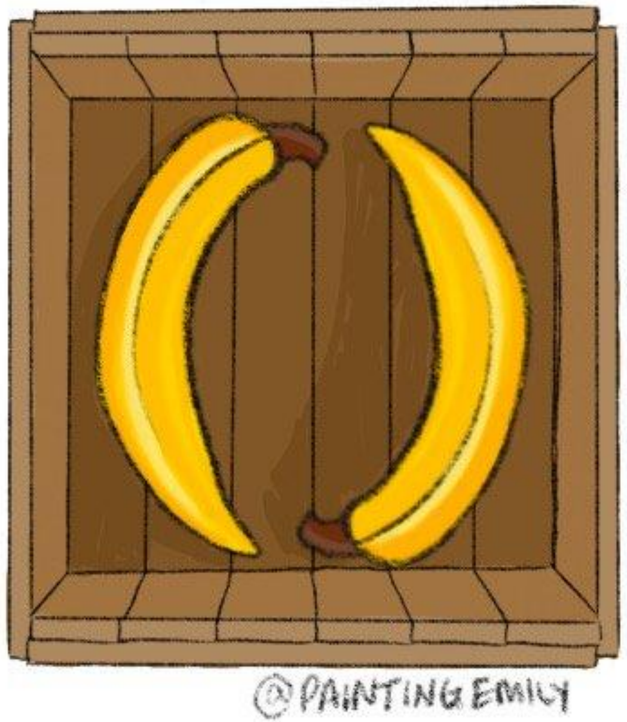
Built-in directive used to manage input value and event

```
<input type="text" [(ngModel)]="prod.price" />
```

Define a binding that detects changes and updates the property value

Two-Way Binding

[(ngModel)] = “bananas in a box” syntax



Built-in Directives

- Angular has built-in directives such as **ngFor** and **ngIf**
- Directives don’t have views associated with them

* = Structural directive that changes the DOM

```
<div *ngIf="userList | async as users; else loading">
   <ul>
      <li *ngFor="let user of users; count as count">
         {{ user.name }}
      </li>
   </ul>
   <div>{{count}} total users</div>
</div>
<ng-template #loading>Loading users...</ng-template>
```

Built-in Pipes

- Angular apps can use Pipes to filter and format data
- Several built-in pipes
  - uppercase, lowercase, slice, date, currency, async, json

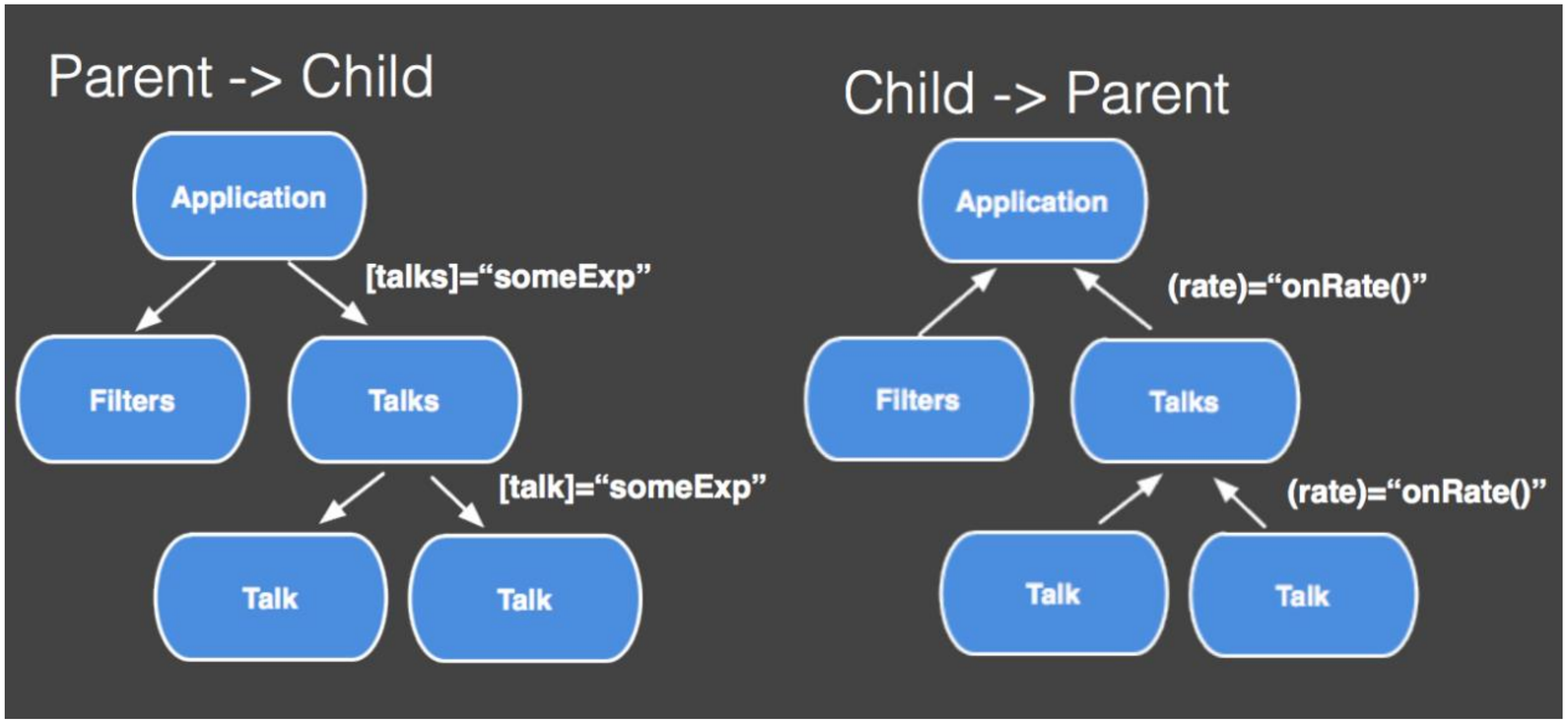
```
<td>{{ customer.orderTotal | currency:'CAD':true:'2.1-3' }}</td>
```

Format as currency

DEMONSTRATION / CODE ALONG

Components' Interactions

- **Input** and **Output** properties (via decorators)



Services

Services ?

- Classes that perform specific set of functionalities (business rules, calculations, Ajax calls, etc.)
- Implemented as singleton (per injector)
- They should be reusable
- Independent from the components
- A Service can consume other Services using dependency injection
- A Service can be used by any Components of a module using dependency injection

Creating a Service

product.service.ts

```
import { Injectable } from '@angular/core';
Import { Http } from '@angular/http';

@Injectable()
export class ProductService {
   constructor(private http: Http) { }
   getProducts() { … }
}
```

@Injectable decorator must be applied if the service has objects injected into it

Injected at runtime

Using a Service

product-list.component.ts

```
@Component({
    selector: 'products',
    templateUrl: 'product-list.component.html',
    providers: [ProductService]
})
export class ProductListComponent {
    products: Product[];
    constructor(private productService: ProductService) { }
    ngOnInit() {
      this.productService.getProducts()
              .subscribe((data: Product[]) => {
                  this.products = data;
              });
    }
}
```

Services must be "provided" to components (or modules…)

ProductService gets injected at runtime

DEMONSTRATION / CODE ALONG

Reactive programming with RxJS

Reactive programming?

- **Reactive programming** is an approach that models data that changes over time.

- Non-reactive Example

```
a = 1;
b = a + 1;

a = 2;

print a; ---> 2
print b; ---> 2
```

- Reactive Example

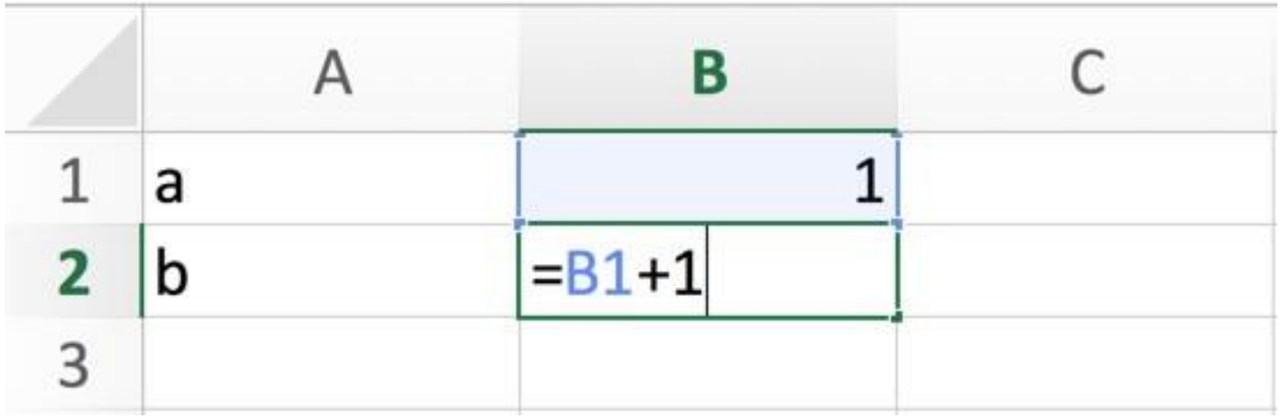
```
a = 1;
b = a + 1;

a = 2;

print a; ---> 2
print b; ---> 3
```

Reactive programming?

- You've done it before!
- Microsoft Excel is a reactive programming interface!



b = a + 1, in Excel

Reactive programming?

- http://reactivex.io/
- **RxJS** is the javascript version
- Technology from Microsoft, Open Source
- Observable pattern
- Asynchronous programming

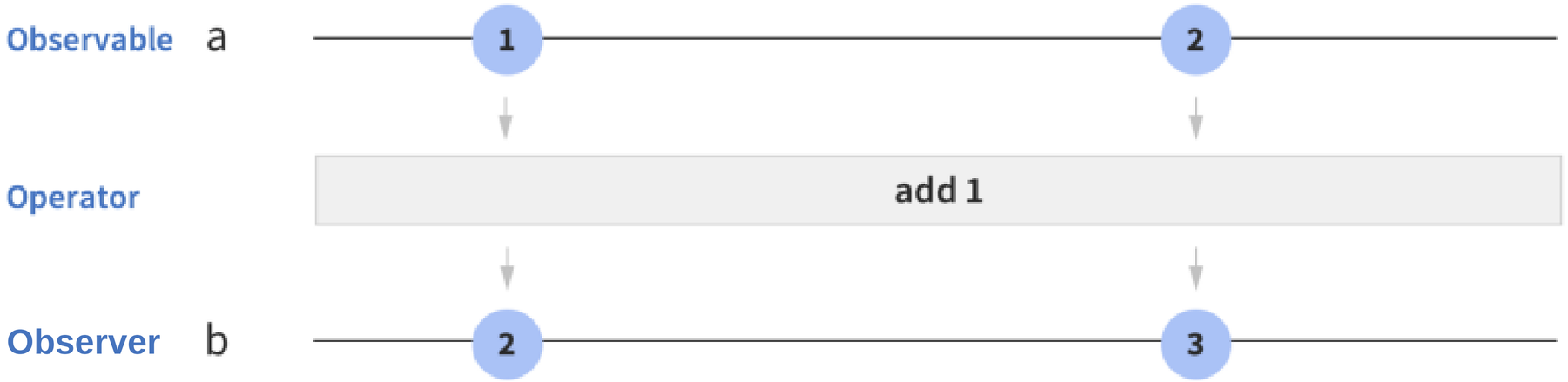
Reactive programming with RxJS

- Asynchronous data streams represented with **Observables**.
- Apply many **Operators** and subscribe with **Observers**.

RxJS = Observables + Operators

Reactive programming with RxJS

- Marbles diagram



Reactive syntax

- Expose an **observable** in a service

DataService

```
GetProducts() : Observable {
   return this
        .http.get(“http://api...”)
        .map(res => res.json();
}
```

- **Subscribe** to it in a component and react

ProductListComponent

```
dataService
    .GetProducts()
    .subscribe(results => {
        this.products = results;
    });
```

RxJS Operators

- Map
- Merge
- Concat
- First, Last, Skip
- Count, Average, Min, Max
- …

http://rxmarbles.com/

DEMONSTRATION / CODE ALONG

HTTP

communication with servers

Angular HTTP

- RESTful services can be called using Http (import **HttpModule** from @angular/http)
- Standard **get, put, post, delete** functions are supported
- Can use **RxJS Observables (**or **Promises)** for **async** operations

Angular HTTP

product.service.ts

```
import { Http, Response } from '@angular/http';
import { Observable } from 'rxjs/Observable';
import 'rxjs/Rx';

@Injectable()
export class ProductService {
    constructor(private http: Http) { }

    getProducts() : Observable<Product[]> {
        return this.http.get('http://api.products.com')
                        .map((response: Response) => response.json())
                        .catch(this.handleError);
    }
}
```

Imports

Inject Http service

Returns an Observable

Angular HTTP

product-list.component.ts

```
@Component(
    { templateUrl: 'product-list.component.html'}
)
export class ProductListComponent implements OnInit {
    public products:Product[] = [];
    constructor(private productService: ProductService) {}

    ngOnInit() {
            this.productService.getProducts()
                .subscribe(
                    data => this.products = data,
                    error => console.log(error)
                );
    }
}
```

Subscribe to the Observable

DEMONSTRATION / CODE ALONG

ngModules

Angular Modules

- Allows grouping of objects (components, services, ...) to obtain a modular app
- Benefits:
  - Code organization
  - Lazy loading
  - Ahead of Time (AoT) Compilation: performances
  - Code optimization: Tree shaking

Module Angular

app.module.ts

```
@NgModule({
  declarations: [ Component1, Directive1, Pipe1],
  imports: [ CommonModule ],
  exports: [ Component1 ],
  providers: [ Service1 ],
  bootstrap: [ AppComponent ]
})
class MyAppModule {}
```

declare all the objects (components, directives, pipes) of the module

everything from the imported modules, declared as export, will be available

objects visible to others modules

classes registered to the app injector as singleton, so available in all modules*

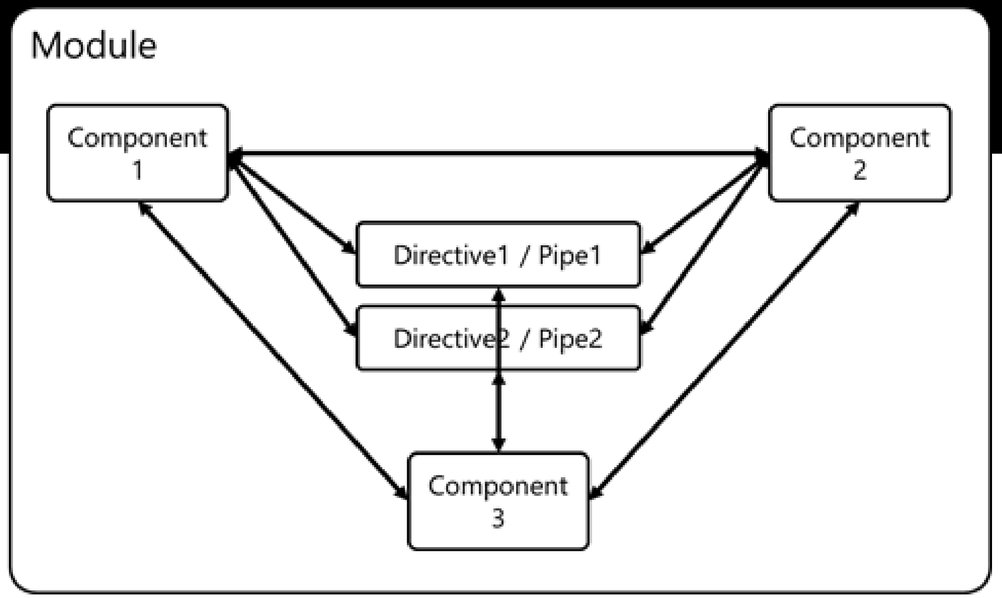
component to launch first, only in root module

* Except for lazy loaded module (via router) which have their own Injector

Module Angular

```
@NgModule({
   declarations: [Component1, Component2, Component3, Directive1,
Directive2, Pipe1, Pipe2]
})
class MyAppModule {}
```

Inside a module all the objects declared know each other.

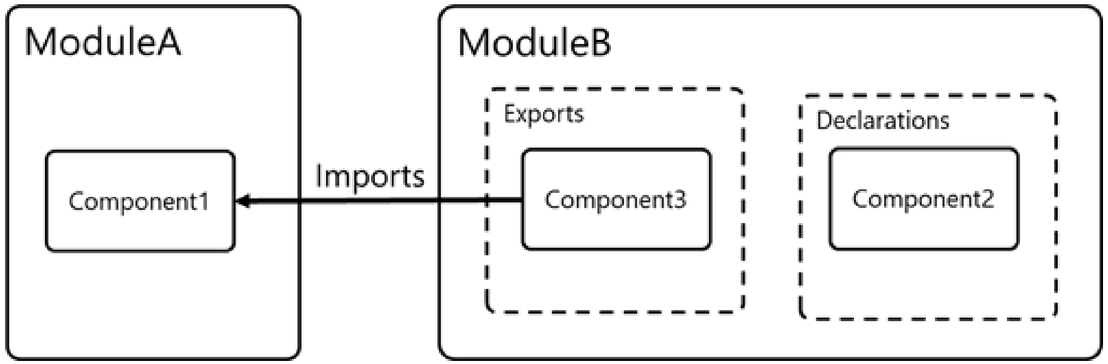


Module Angular

```
@NgModule({
   declarations: [Component1],
   imports:      [ModuleB]
})
class ModuleA {}
```

```
@NgModule({
   declarations: [Component2, Component3],
   exports:      [Component3]
})
class ModuleB {}
```

Component1 can use Component3 in its template but can’t use Component2



DEMONSTRATION / CODE ALONG

Release into production

Just-In-Time vs Ahead-of-Time Compilation

- **Just-In-Time** compilation (dev mode)

  ng serve

  - App compiled in the browser, at runtime
  - Need to embed the Angular compiler
  - Discover template errors at runtime
  - Only mode supported in Angular 1
- **Ahead-of-Time** compilation (prod mode)

  ng serve --prod

  - Everything compiled at build time
  - Catch template errors early
  - No need to download the Angular compiler

Ahead-of-Time Compilation

- Full pre-compilation of the app
  - Components template pre-compiled to Js
  - Fast initial rendering
- Remove Angular compiler from bundle!

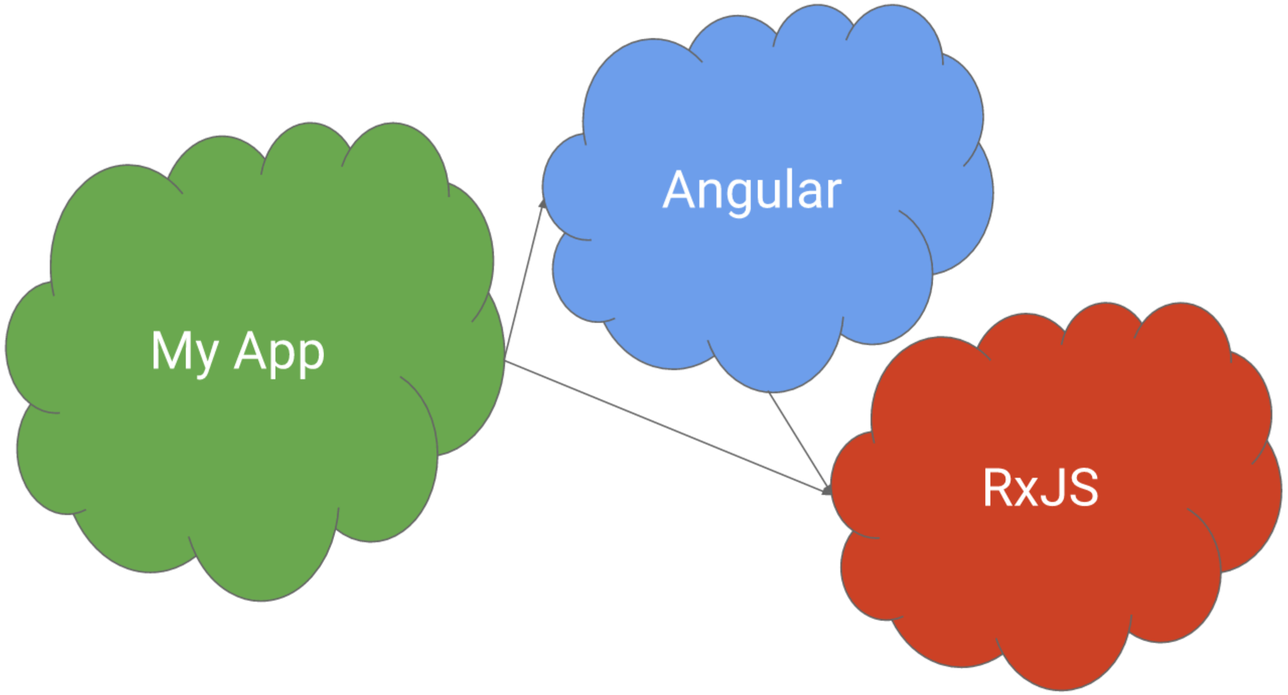
Huge app startup improvement

Huge runtime performance improvements

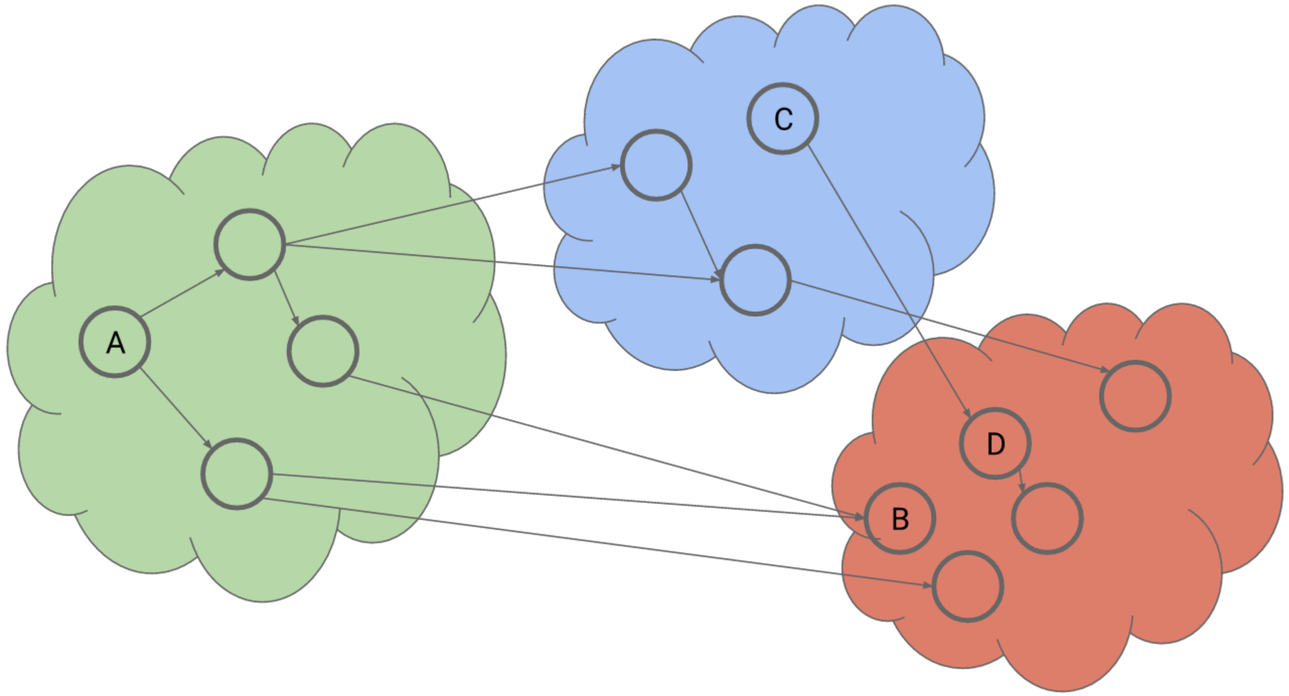
Tree Shaking

- Remove unused code from bundles

Tree Shaking



Tree Shaking

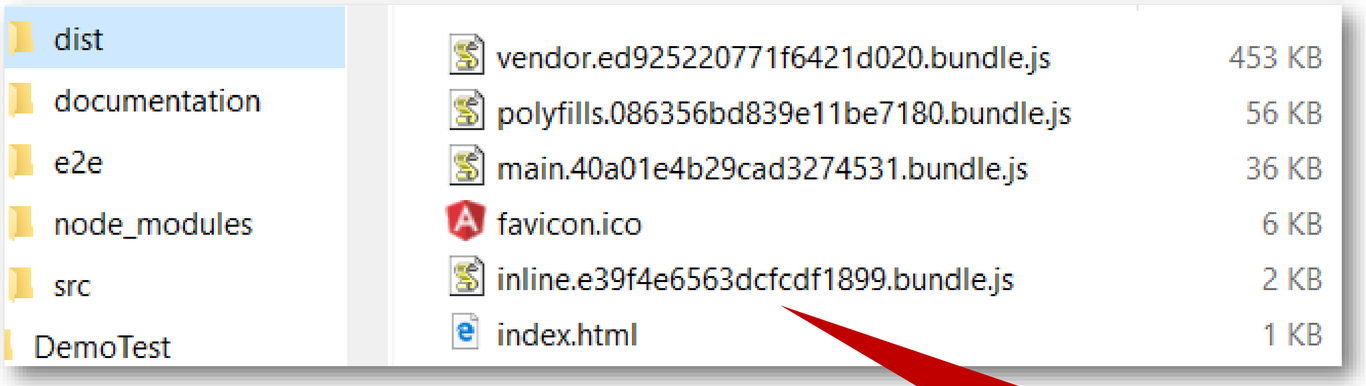


- We can safely remove C, D, …

Deploy to production

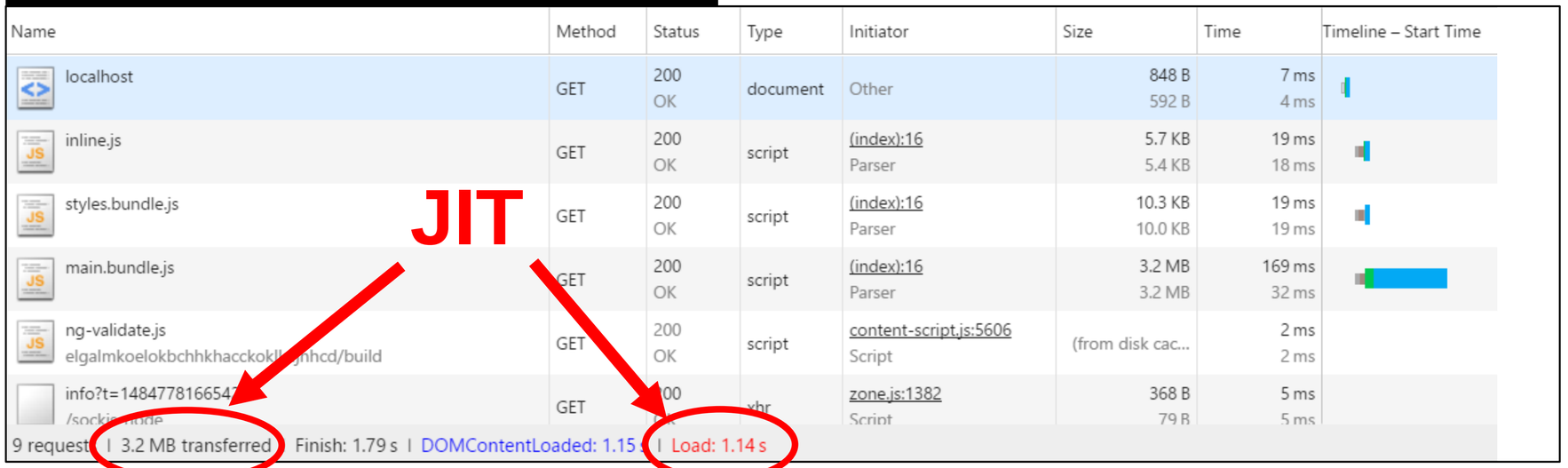
- Generate your optimized bundles

```
> ng build --prod
```

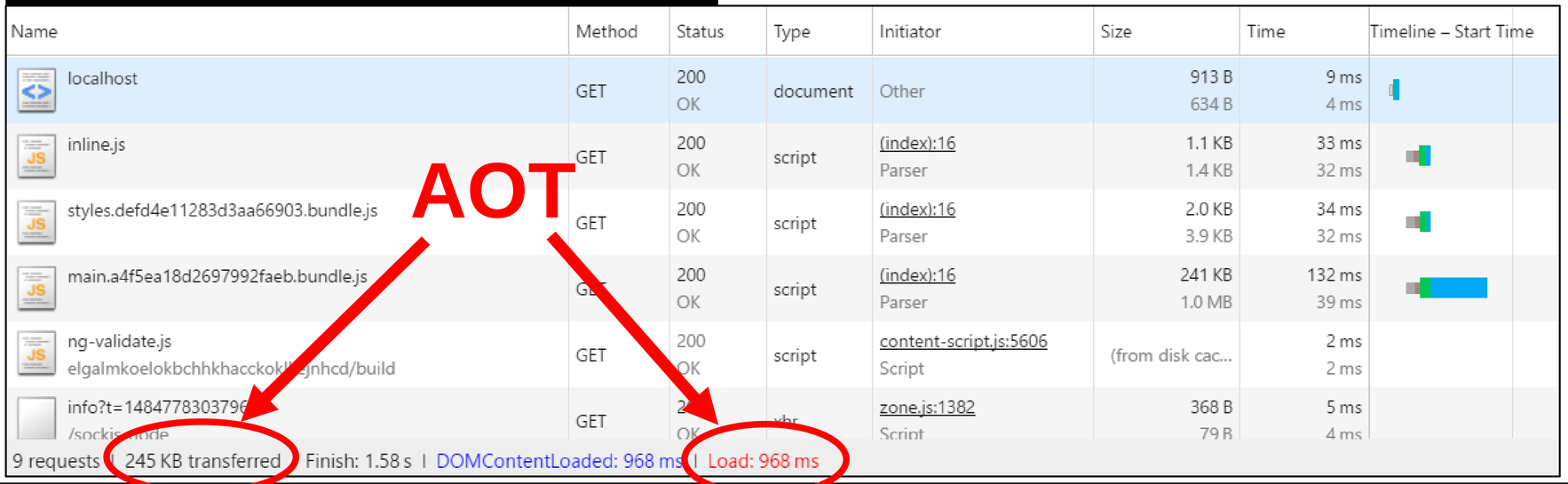


DEMONSTRATION / CODE ALONG

> ng serve



> ng serve --prod



CLI with WebPack

Pre-compilation (AoT)
- Pre-compile templates
- Do not include Ng compiler in bundle
- No source maps

Bundling
- Merge all Js files to one big package

Tree Shaking
- Drop unused exports
- Code compression

JavaScript minifying

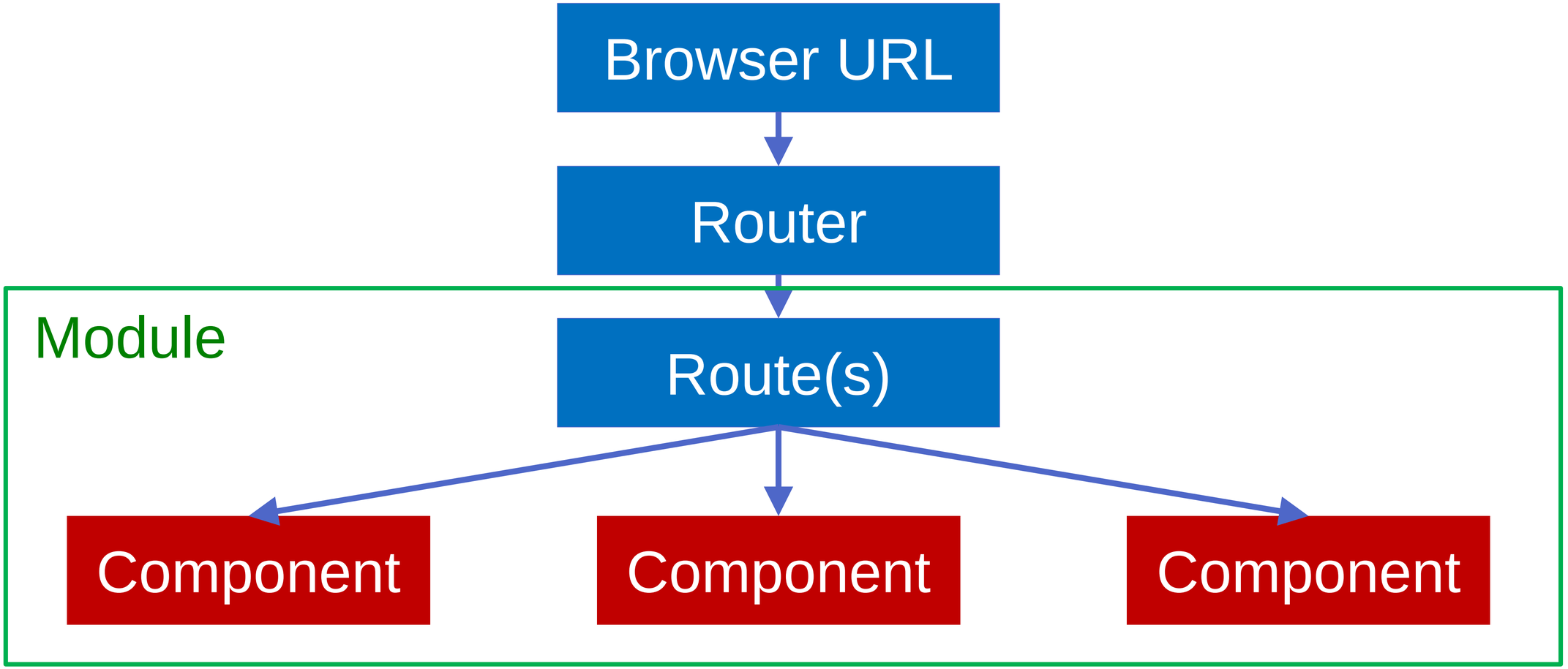
Gzip ← with gzip compression activated on your web server

```
> ng build --prod
```

Bundles can be up to 36x smaller!

Routing

Routing ?


Browser URL
Router
Module
Route(s)
Component
Component
Component

Routes Configuration

- Define the **routes** in a dedicated file

app.routing.ts

```
import { Routes, RouterModule } from '@angular/router';
import { ProductListComponent } from './product-list.component';
import { ContactComponent } from './contact.component';

const appRoutes: Routes = [
  { path: 'products', component: ProductListComponent },
  { path: 'contact', component: ContactComponent }
];

export const routing = RouterModule.forRoot(appRoutes);
```

Routes Configuration

- You can also have **child routes**

app.routing.ts

```
export const AppRoutes = [
  { path: 'products',
    children:[
        { path: '', component: ProductListComponent }
        { path: ':id', component: ProductDetailComponent }
    ]
  },
  { path: 'contact', component: ContactComponent }
])
```

Routes Configuration

- Set a **default route**

app.routing.ts

```
export const AppRoutes = [
  { path: '', redirectTo:'/products', pathMatch: 'full' },
  { path: 'products'
    children:[
        { path: '', component: ProductListComponent }
        { path: ':id', component: ProductDetailComponent }
    ]
  },
  { path: 'contact', component: ContactComponent }
])
```

Router Module

- Import the configured RouterModule in the root module

app.module.ts

```
import { AppComponent } from './app/';
import { routing } from './app.routing';

@NgModule({
    declarations: [AppComponent],
    imports: [routing],
    bootstrap: [AppComponent]
})
export class AppModule {
}
```

Router Directives

- Components are loaded into the **<router-outlet></router-outlet>**

Component

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

@Component({
    selector: 'store-app',
    template: `…<router-outlet></router-outlet>…`
  })
export class AppComponent
{
```

Route Navigation

- Navigate between Components using the **RouterLink** directive

Template

```
<nav>
   <a [routerLink]="['products']">Products</a>
   <a [routerLink]="['products',{id:7}]">Product 7</a>
   <a [routerLink]="['contact']">Contact</a>
</nav>
```

- Or by code

```
router.navigate( ['/products', prodId] );
router.navigateByUrl('/products' + prodId);
```

Route Parameters

- Use **ActivatedRoute** to get optional parameters

Component

```
import { ActivatedRoute } from '@angular/router';

constructor(private route: ActivatedRoute) { }

ngOnInit() {
    // + to convert string to int
    let id = + this.route.snapshot.params['id'];
    if (id) {
        this.productsService.getProductById(id).then(data => this.product = data);
    }
}
```

DEMONSTRATION / CODE ALONG

Routes Guards

Decides if a route can be activated
- User logged-in ?

Decides if a route can be deactivated
- Form saved ?

Victor Savkin Follow

Victor makes Angular. He also toys with eclectic programming technologies and obsesses over fonts and ke...

Aug 15 · 7 min read

Angular 2 Router

http://tinyurl.com/zzeevj9

100
PAGES

ENGLISH PDF EPUB MOBI APP

Angular Router

The Complete Authoritative Reference

Victor Savkin

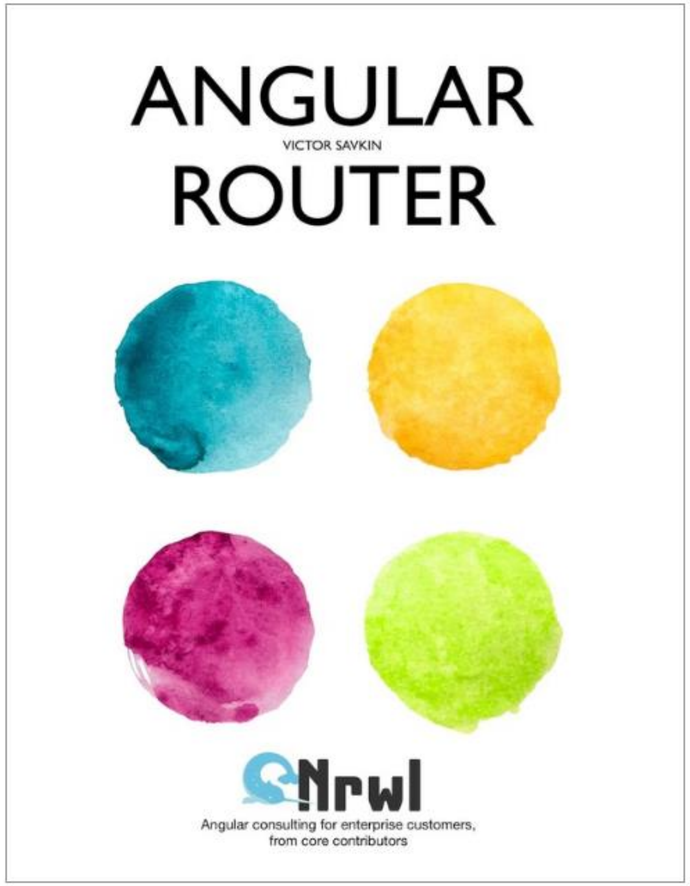
This book is a comprehensive guide to the Angular router written by its designer, who is a co-founder Nrwl - Angular Consulting for enterprise customers, from core contributors.

The book explores the library in depth, including the mental model, design constraints, subtleties of the API.

Reading the book will make you an Angular router expert.

Read Free Sample

Table Of Contents



This book is 100% complete

$20.00 MINIMUM

$25.00 SUGGESTED

YOU PAY

$25.00

AUTHOR EARNS

$22.00

YOU PAY (US$)

$25.00

EU customers: Price excludes VAT. VAT is added during checkout.

Add Ebook to Cart

Add to Wish List

50% OFF

https://leanpub.com/router/c/academy

Lazy Loading

Lazy load your modules!

app.routing.ts

```
const routes: Routes = [
 {path: '', redirectTo:'/home', pathMatch:'full'},
 {path: 'products', loadChildren:'./products/products.module#ProductsModule'},
 {path: 'home', component: HomeComponent},
 {path: 'contact', component: ContactComponent},
];
```

Lazy load your modules!

products.routing.ts

```
const routes: Routes = [
  { path: 'products', component: ProductListComponent },
  { path: 'products/:id', component: ProductDetailComponent }
];
```

products.routing.ts

```
const routes: Routes = [
  { path: '', component: ProductListComponent },
  { path: ':id', component: ProductDetailComponent }
];
```

Lazy load your modules!

app.module.ts

```
@NgModule({
imports:[BrowserModule, FormsModule, HttpModule, routing,
   ProductsModule],
})
export class AppModule { }
```

app.module.ts

```
@NgModule({
imports:[BrowserModule, FormsModule, HttpModule, routing],
})
export class AppModule { }
```

DEMONSTRATION / CODE ALONG

Without lazy loading

Name	Method	Status	Type	Initiator	Size	Time	Timeline – Start Time
home	GET	200 OK	document	Other	913 B 634 B	9 ms 7 ms	
inline.js	GET	200 OK	script	home:16 Parser	1.1 KB 1.4 KB	24 ms 23 ms	
styles.60d4e58a663e12fc2e0d.bundle.js	GET	200 OK	script	home:16 Parser	2.6 KB 6.1 KB	24 ms 23 ms	
main.9c730429f3d3cf39b367.bundle.js	GET	200 OK	script	home:16 Parser	256 KB 1.1 MB	160 ms 32 ms	
ng-validate.js elgalmkoelokbchhkhacckoklkejnhcd/build	GET	200 OK	script	content-script.js:5606 Script	(from disk cac...	3 ms 3 ms	
info?t=1484779353265 /sockjs-node	GET	200 OK	xhr	zone.js:1382 Script	368 B 79 B	6 ms 5 ms	

9 requests | 261 KB transferred | Finish: 1.92 s | DOMContentLoaded: 1.37 s | Load: 1.37 s

With lazy loading

Name	Method	Status	Type	Initiator	Size	Time	Timeline – Start Time
home	GET	200 OK	document	Other	913 B 634 B	9 ms 7 ms	
inline.js	GET	200 OK	script	home:16 Parser	1.1 KB 1.4 KB	25 ms 22 ms	
styles.60d4e58a663e12fc2e0d.bundle.js	GET	200 OK	script	home:16 Parser	2.6 KB 6.0 KB	27 ms 22 ms	
main.4600d397b418894c4566.bundle.js	GET	200 OK	script	home:16 Parser	227 KB 949 KB	142 ms 29 ms	
ng-validate.js elgalmkoelokbchhkhacckoklkejnhcd/build	GET	200 OK	script	content-script.js:5606 Script	(from disk cac...	2 ms 2 ms	
info?t=1484779591298 /sockjs-node	GET	200 OK	xhr	zone.js:1382 Script	368 B 79 B	5 ms 4 ms	

9 requests | 232 KB transferred | Finish: 1.67 s | DOMContentLoaded: 961 ms | Load: 961 ms

Resources

- Angular Observable Data Services http://tinyurl.com/hennfnw, http://tinyurl.com/gp6ff2x
- Material http://tinyurl.com/z9q2wn5
- SEO http://tinyurl.com/hvvedsd, https://prerender.io/
- ngModules, AOT, Lazy Loading http://tinyurl.com/ha2bdyl
- Router http://tinyurl.com/jfyzccu, http://tinyurl.com/gr8r9ny
- Advanced Styling Guide http://tinyurl.com/jrn28pf
- Components Lifecycle http://tinyurl.com/h826r7k
- Internationalization http://tinyurl.com/h9xqjqe

Extremely high quality articles!

- https://blog.nrwl.io
- https://blog.thoughtram.io
- http://blog.mgechev.com
- www.syntaxsuccess.com/angular-2-articles
- http://blog.angular-university.io
- https://coryrylan.com

Certificate!

PRAIRIE
ANGULAR DEV CON WORKSHOP
CLOUD | MOBILE | WEB | DEV

Write your name here

has successfully completed the Angular Workshop at Prairie Dev Con 2017

ANGULAR
ACADEMY
www.angularacademy.ca

INSTRUCTOR: Laurent DUVEAU

DATE: 2017-06-05

ANGULAR WORKSHOP

Thank you!

http://angular.ac/comment