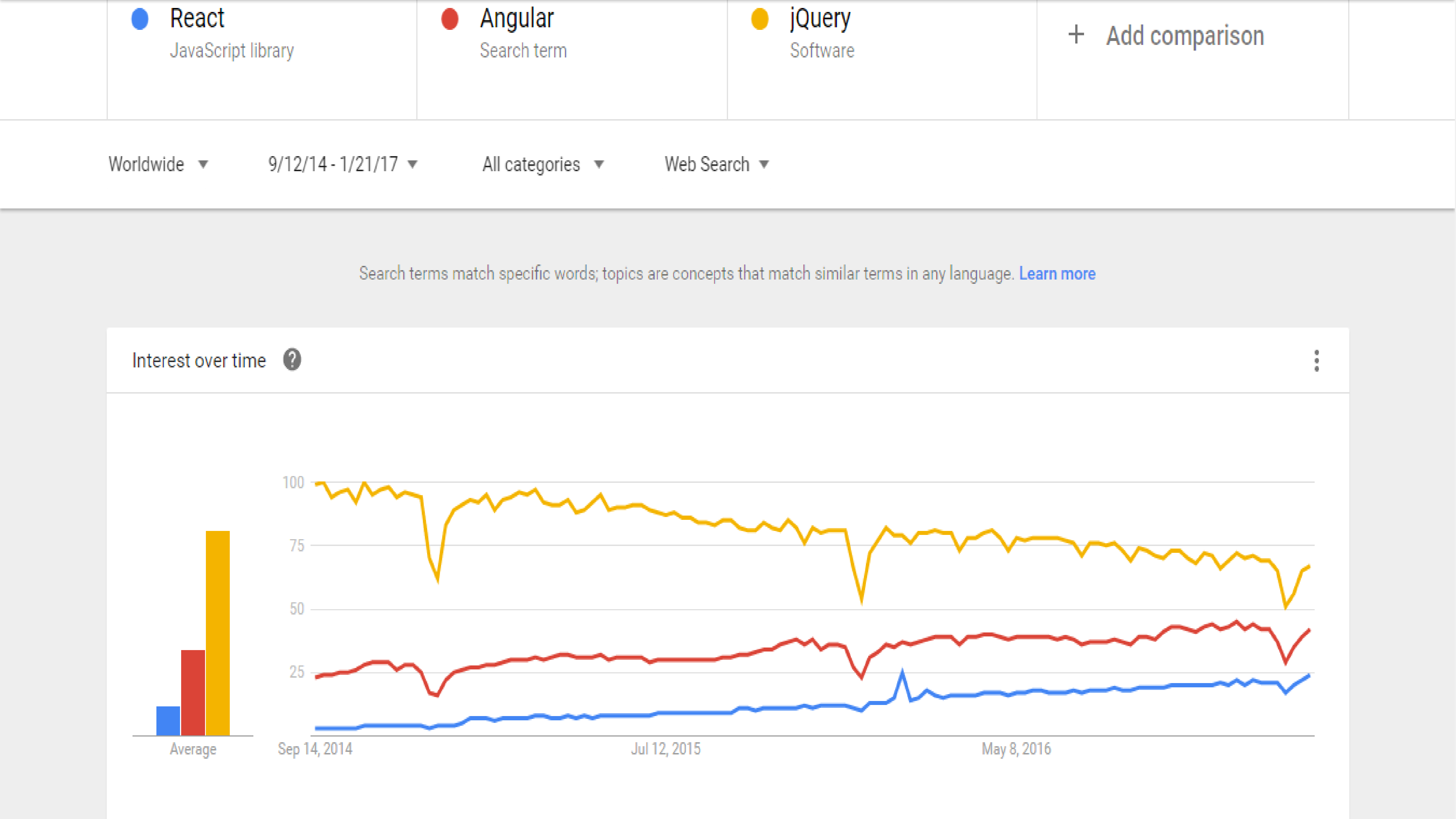
BUILD ANGULAR APPS

Pavel Kyurkchiev
PhD. candidate
@pkyurkchiev
https://github.com/pkyurkchiev

Angular 2

Getting Started

100
120
140
160
180
200
220
240
260
280
300



● React
JavaScript library

● Angular
Search term

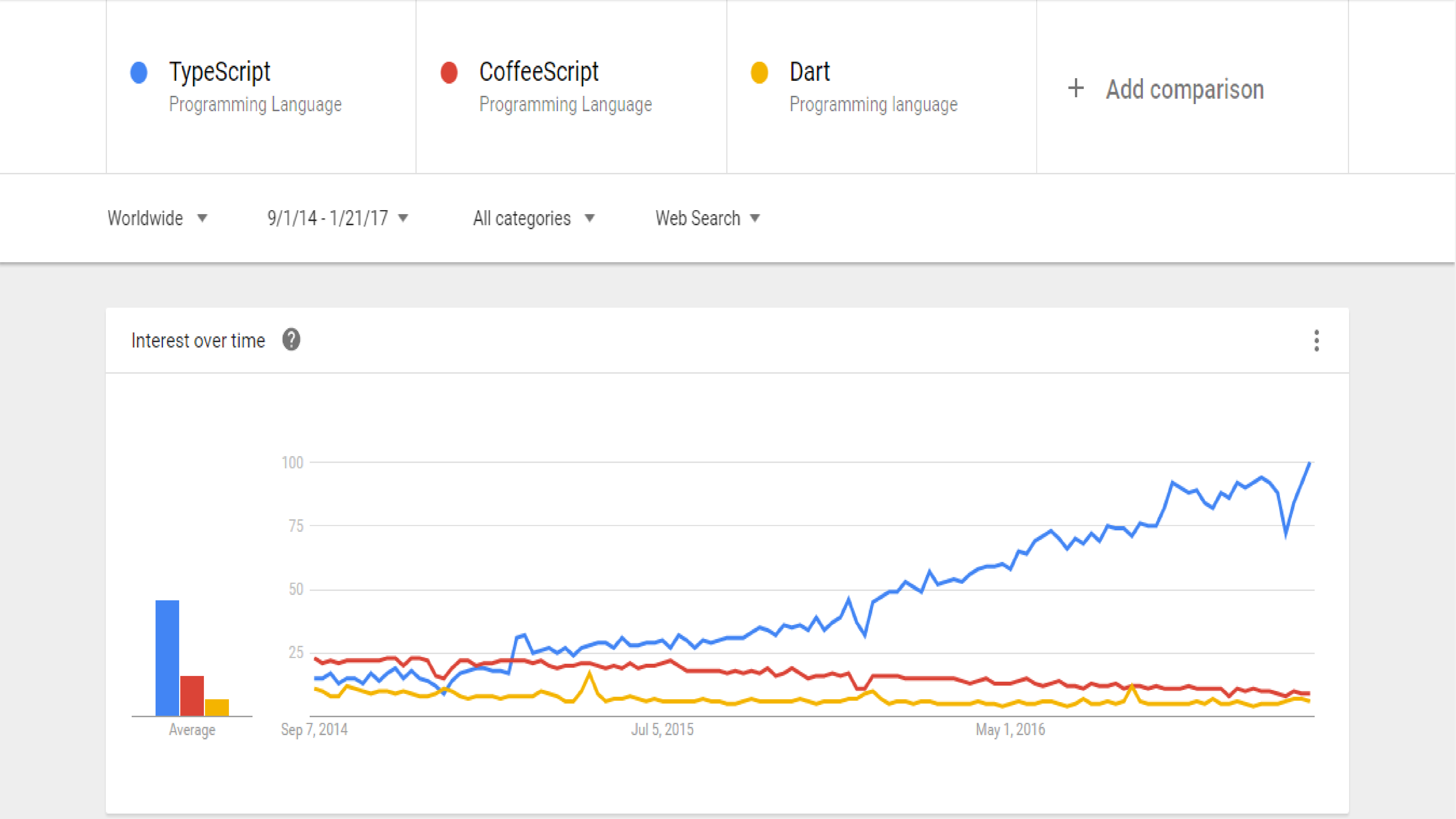
● jQuery
Software

+ Add comparison

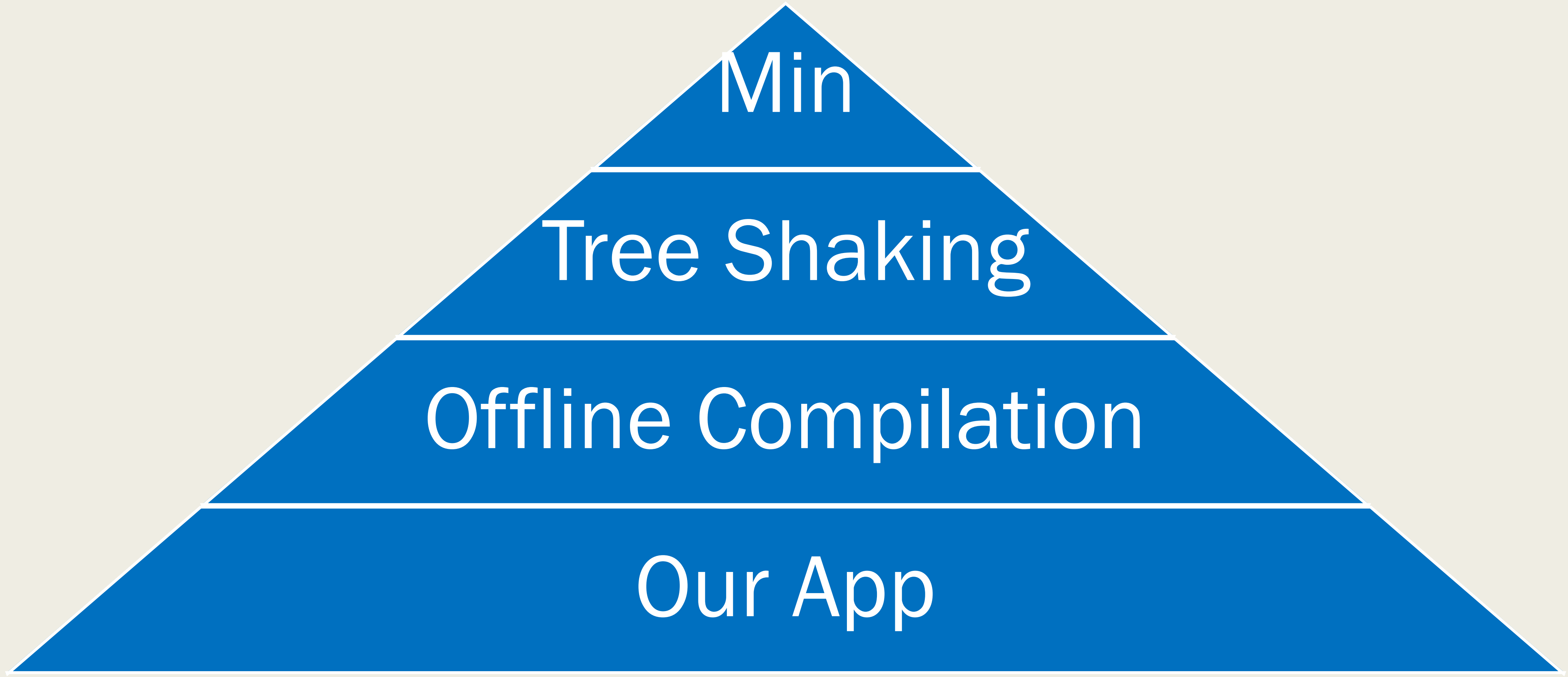
Worldwide ▼ 9/12/14 - 1/21/17 ▼ All categories ▼ Web Search ▼

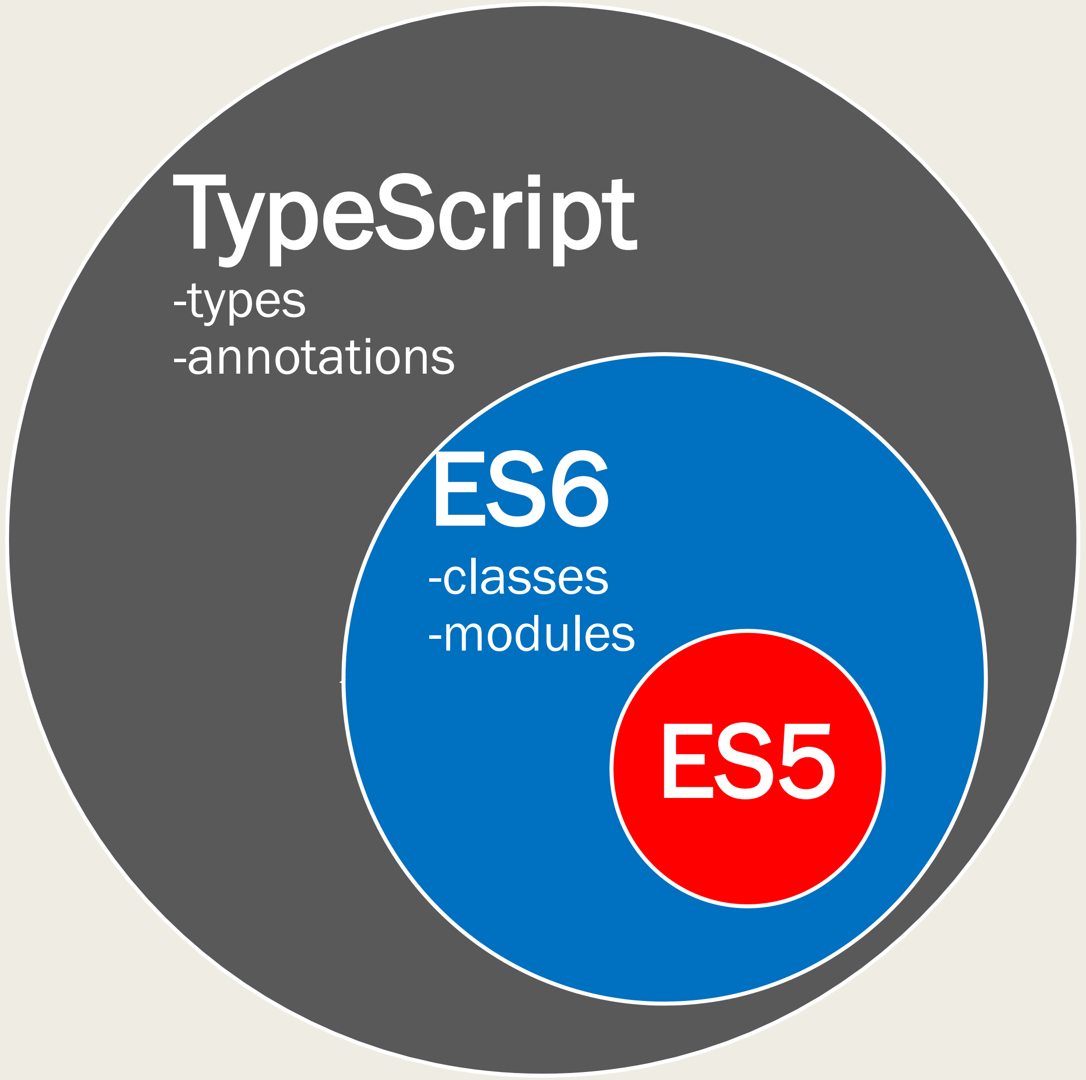
Search terms match specific words; topics are concepts that match similar terms in any language. Learn more

Interest over time


● TypeScript
Programming Language
● CoffeeScript
Programming Language
● Dart
Programming language
+ Add comparison
Worldwide ▼
9/1/14 - 1/21/17 ▼
All categories ▼
Web Search ▼
Interest over time
100
75
50
25
Average
Sep 7, 2014
Jul 5, 2015
May 1, 2016

Deliver Optimized and Smaller Code to the Clients




TypeScript
-types
-annotations
ES6
-classes
-modules
ES5

Familiar Concepts

Services

Components

HTTP

Events

Data Binding

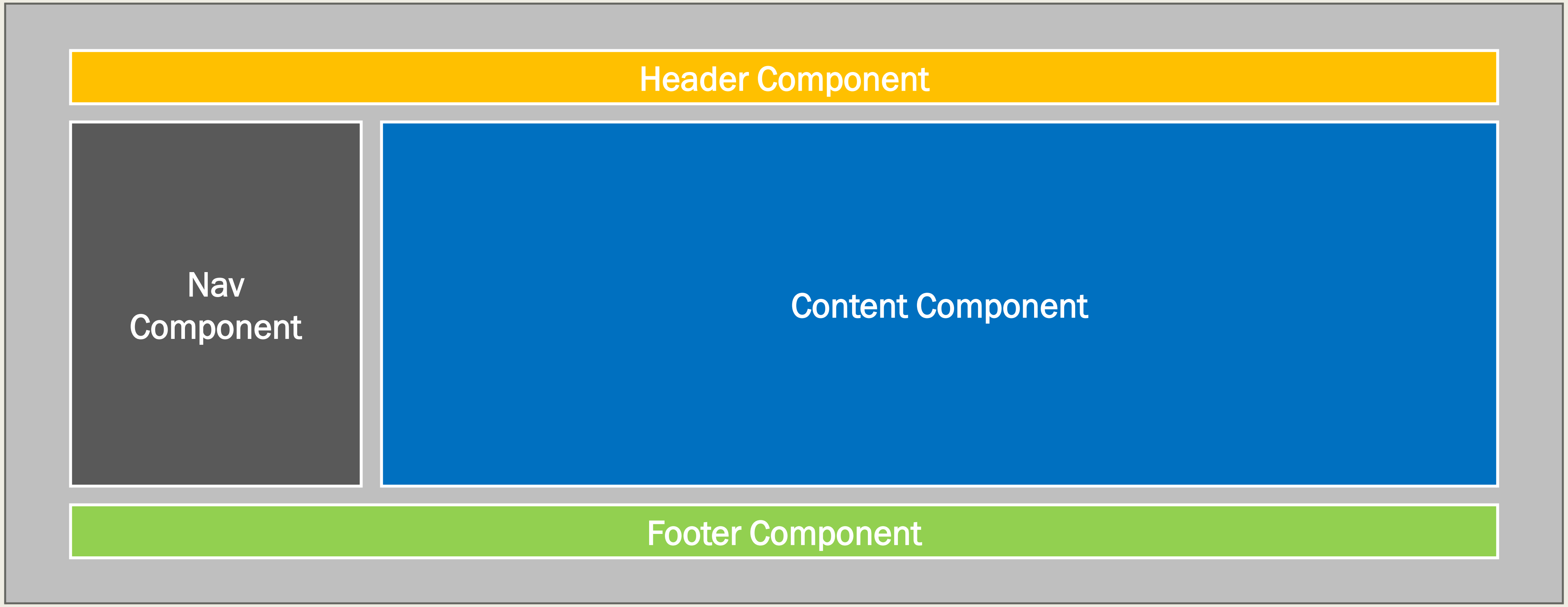
Components

- A component contains application logic that controls a region of the user interface that we call a view.

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

import {Vehicle} from './vehicle.service';

@Component({
    moduleId: module.id,
    selector: 'story-vehicles',
    templateUrl: 'vehicles.component.html'
})

export class VehicleListComponent {
    vehicles: Vehicle[];
}
```

Anatomy of Component

App from Components



DEMO COMPONENTS

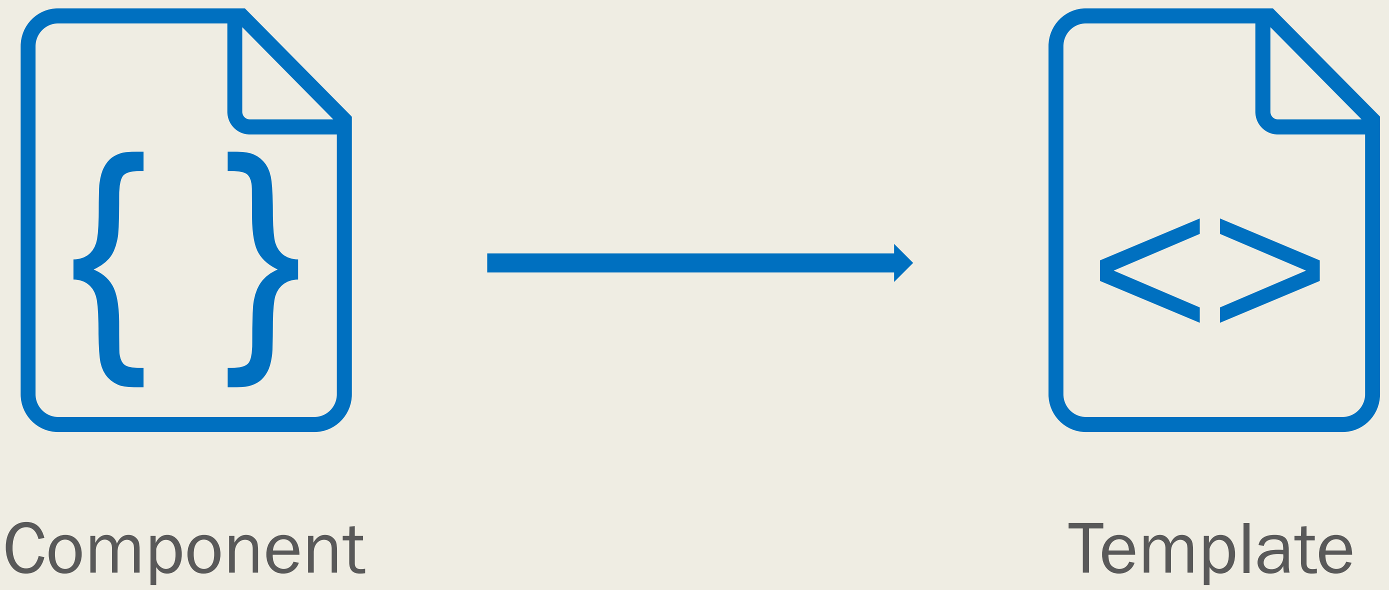
Templates

- Templates are mostly HTML, with a little help from Angular. They tell Angular hot to render the Component.

```
<ul>
    <li *ngFor="let character of characters">
        {{ character.name }}
    </li>
</ul>

<my-character *ngIf="selectedCharacter"
 [character]="selectedCharacter"></my-character>
```

Templates

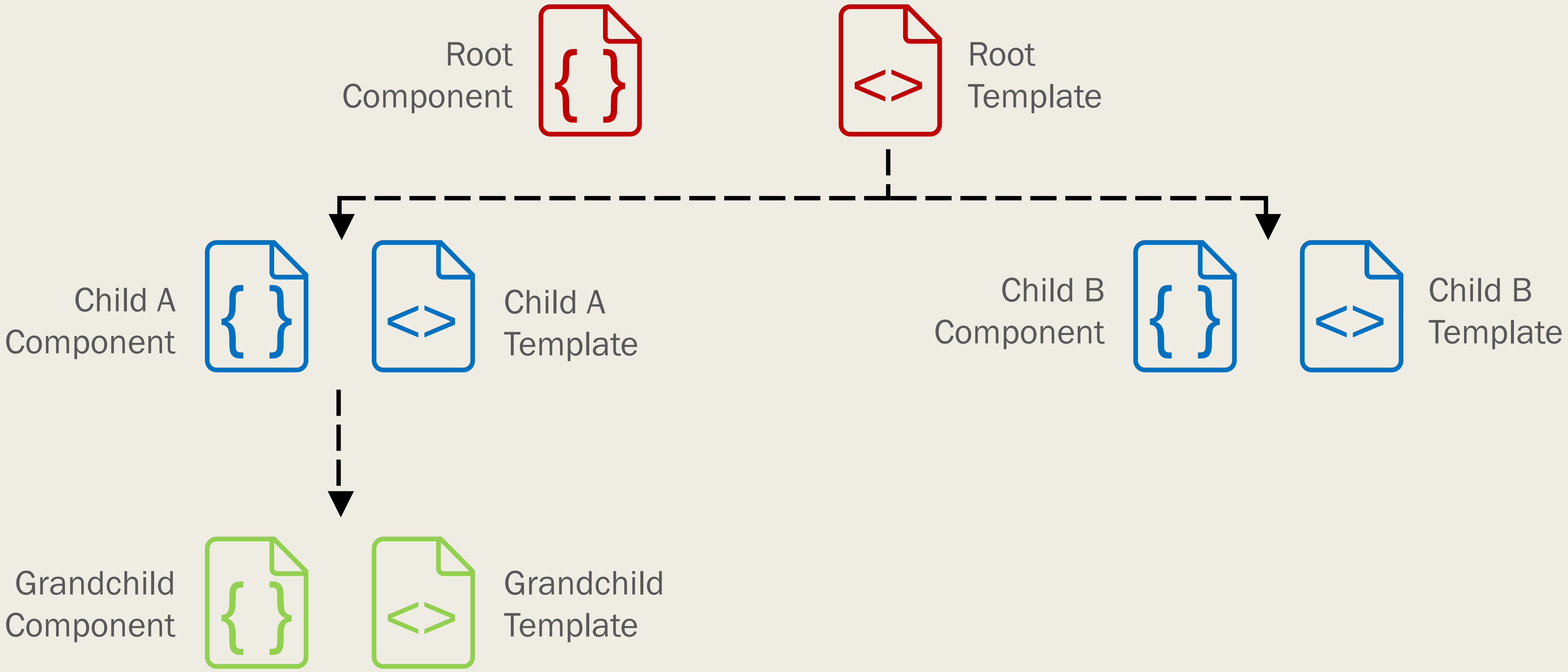
Connecting the Component to its Template



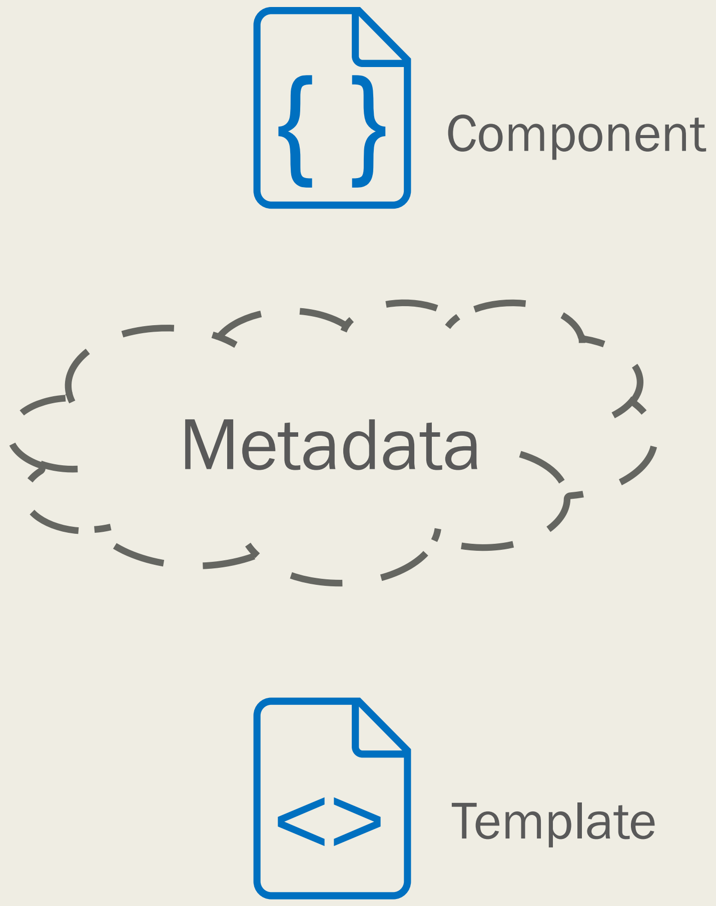
```
@Component({
    moduleId: module.id,
    selector: 'story-vehicles',
    templateUrl: 'vehicles.component.html'
})

export class VehicleListComponent { }
```

Linked Templates with Component

Templates Contain Other Components



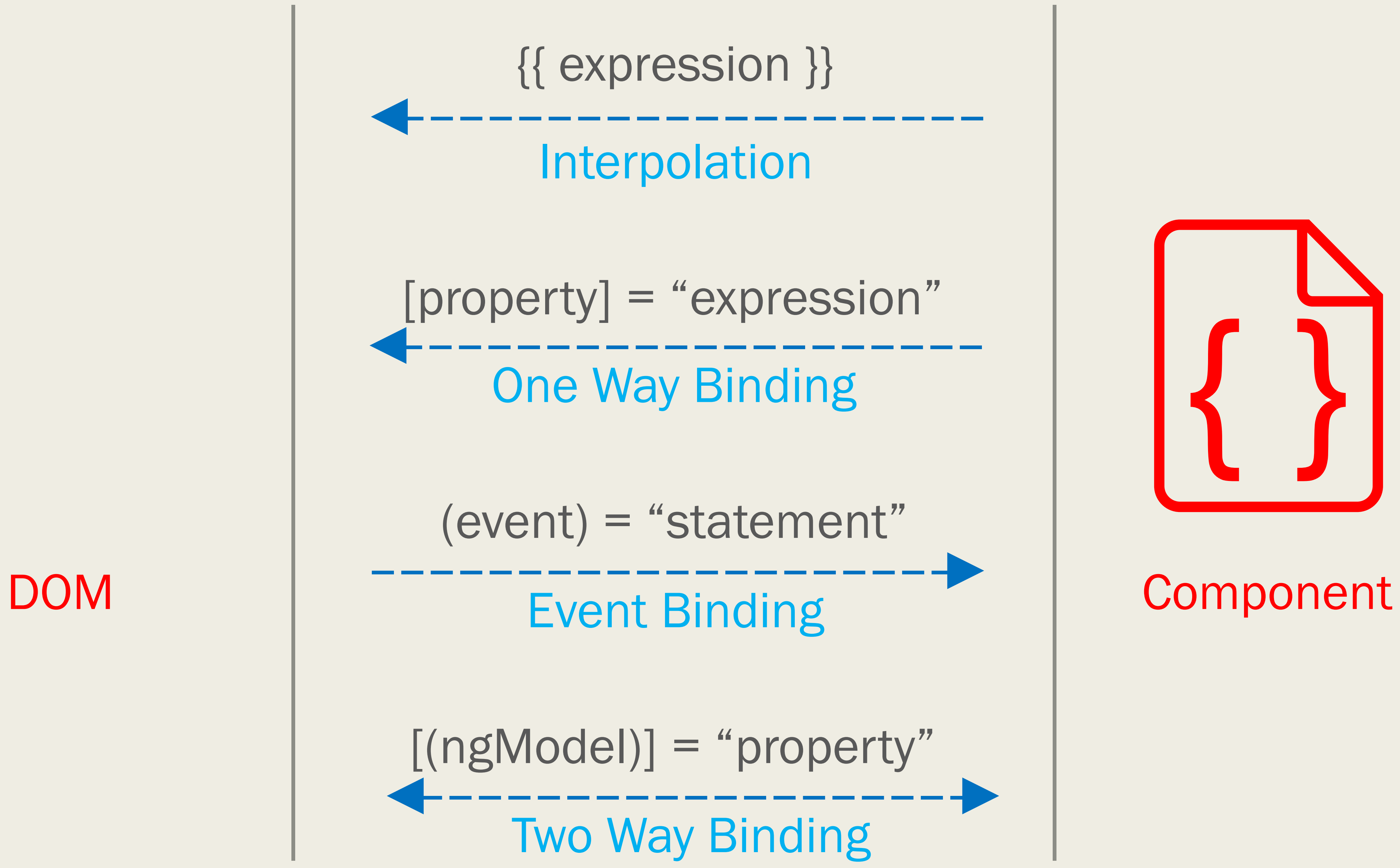
Metadata links the Template to the Component



DEMO TEMPLATES

Data Binding

- We use data binding to help coordinate communication between a Component and its Template.


DOM
{{ expression }}
Interpolation
[property] = “expression”
One Way Binding
(event) = “statement”
Event Binding
[(ngModel)] = “property”
Two Way Binding
Component

Benefits of Angular 2 Unidirectional Data Flow

Easies 3-rd party widget integration

No more $apple

No more repeated digest cycles

No more watchers

No more performance issues with digest cycle and watcher limits

```
<h3>Vehicle: {{vehicle.name}}</h3>
<div>
    <img src="{{vehicle.imageUrl}}" />
    <a href="{{vehicle.wikiLink}}"Wiki></a>
</div>
```

Interpolation - one way in

```
<img [src]="vehicle.imageUrl" />

<vehicle-detail [vehicle]="currentVehicle"></vehicle-detail>

<div [ngClass]="{selected: isSelected}">X-Wing</div>
```

Property Binding - one way in

Event Binding

Using the () to send events from the Template to the Component

```
<button (click)="save()">Save</button>

<vehicle-detail (changed)="vehicleChanged()"></vehicle-detail>
```

Event Binding – one way to the component

Two Way Binding

[()] sends a value from Component to Template, and sends changes in the Template to the Component

```
<input [(ngModel)]="vehicle.name" />
```

To Way Binding – value in, value out

DEMO DATA BINDING

Built-In Directives

- When Angular renders templates, it transforms the DOM according to instructions from Directives

```
<div *ngIf="currentVehicle">
    You selected {{currentVehicle.name}}
</div>
```

*ngIf – conditional template

Conditionally removes elements from the DOM
Structural directive
Use [style.visibility]=“isVisible()” to hide

```
<ul>
    <li *ngFor="let character of characters">
        {{ character.name }}
    </li>
</ul>
```

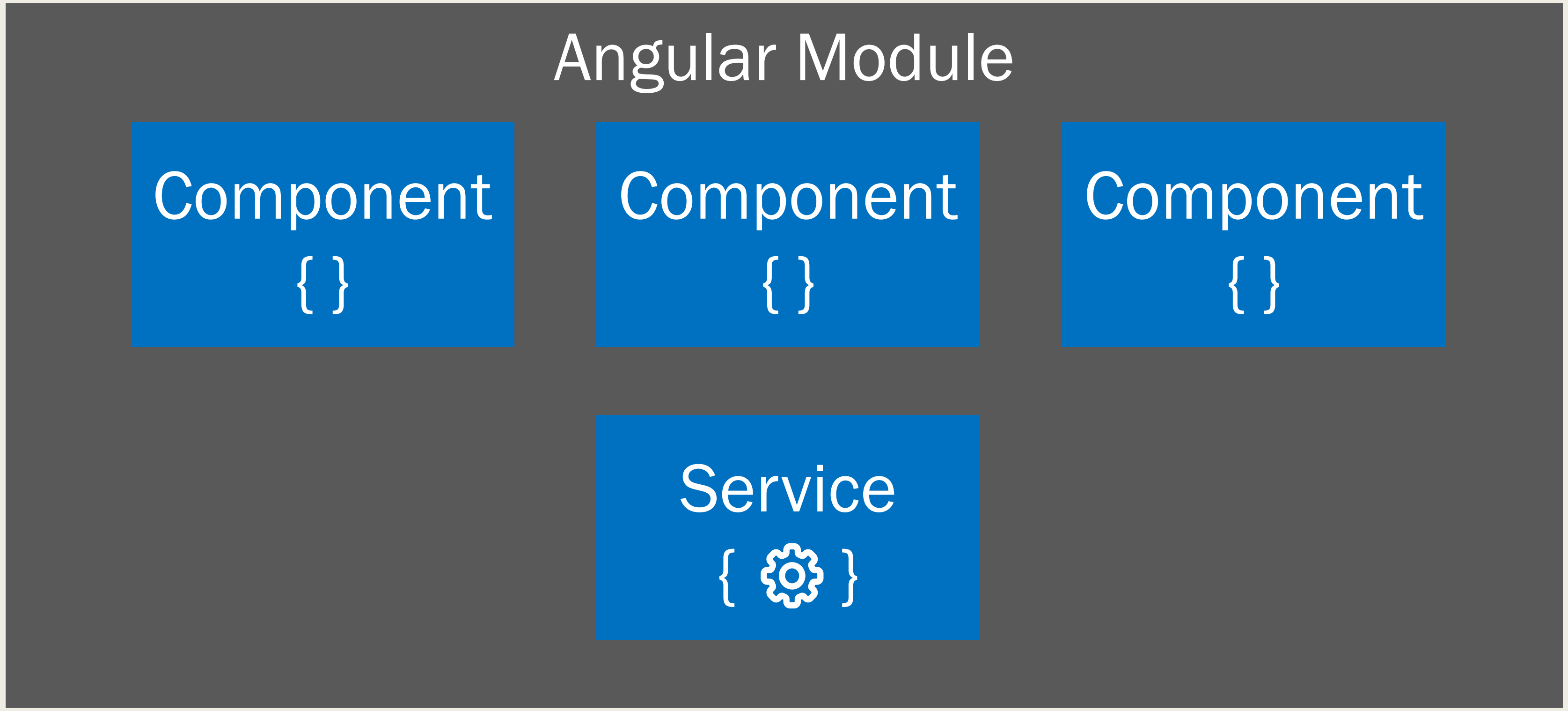
*ngFor – repeating a template

Structural directive
Show an element n number of times

Angular Modules

- We use NgModule to organize our application into cohesive blocks of related functionality.

Angular Modules Organize Functionality



Angular Module is a class decorated by @NgModule

Roles of Angular Models

- Import other Angular Modules
- Identify Components Pipes, and Directives
- Export its features
- Provide services to injectors
- Can be eagerly or lazily loaded

Every app begins with one Angular Module

```
@NgModule({
    imports: [
        BrowserModule, FormsModule
    ],
    declarations: [
        VehiclesComponent
    ],
    providers: [
        VehicleService
    ],
    bootstrap: [VehiclesComponent],
})
export class AppModule { }
```

Root Angular Module

Services

Service provides everything our application needs. It often shares data or functions between other Angular features.

Services

Angular 1

- Factories
- Services
- Providers
- Constants
- Values

Angular 2

- Class

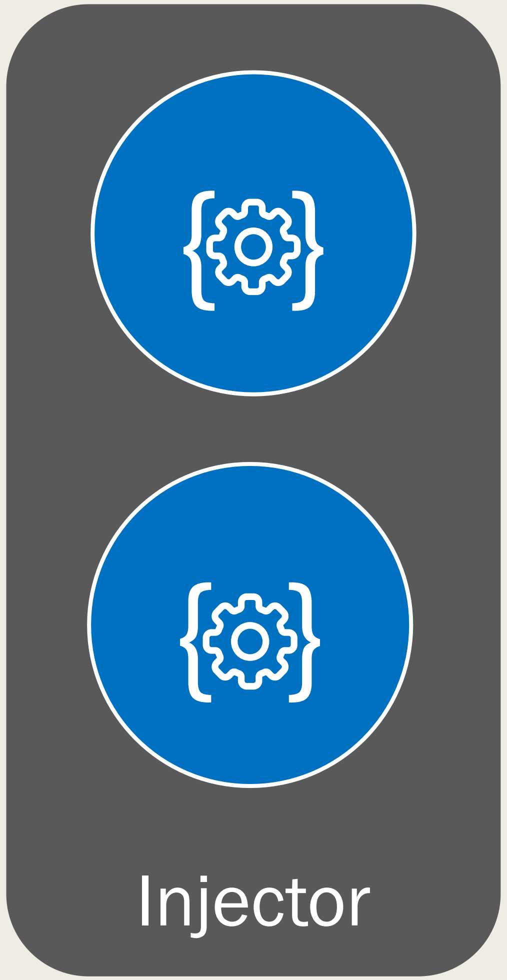
```
@Injectable()
export class VehicleService{
    getVehicles(){
        return [
          new Vehicle(1, 'Falcon'),
          new Vehicle(2, 'X-Wing Fighter')
        ];
    }
}
```

Service

DEMO MODULES

Dependency Injection

Dependency Injection is how we provide an instance of a class to another Angular feature.



```
export class VehicleListComponent {
    vehicles: Vehicle[];

    constructor(private vehicleService: VehicleService) { }
}
```

Injecting a Service into a Component

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

    getVehicles(){
        return this.http.get(vehiclerUrl)
        .map((res: Response) => res.json().data);
    }
}
```

Injecting a Service into a Service

When we inject a service, Angular searches the appropriate injectors for it