Angular 2 Overview

Jesse Warden | Accenture | OpenSlava 2015

What

- Angular 2 Overview
- Programming Languages
- Highlights

Angular 2 Overview

- From Google & Microsoft
- Open Source Community
- It’s in Alpha

Alpha?

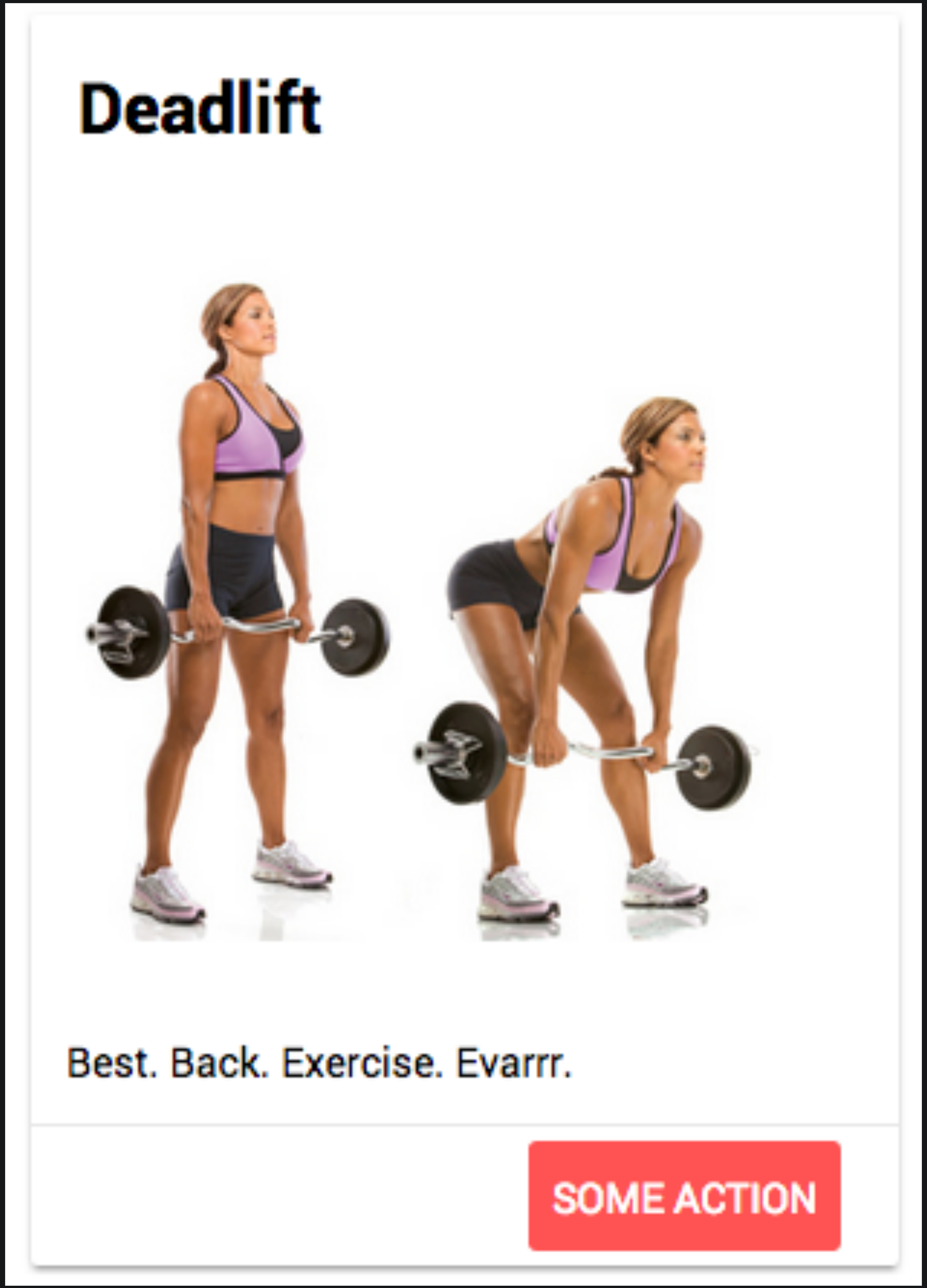
- Developer Preview
- You can play with right now at angular.io
- API Keeps changing…

Embraces Web Standards

- ShadowDOM
- WebWorkers
- Native

ShadowDOM

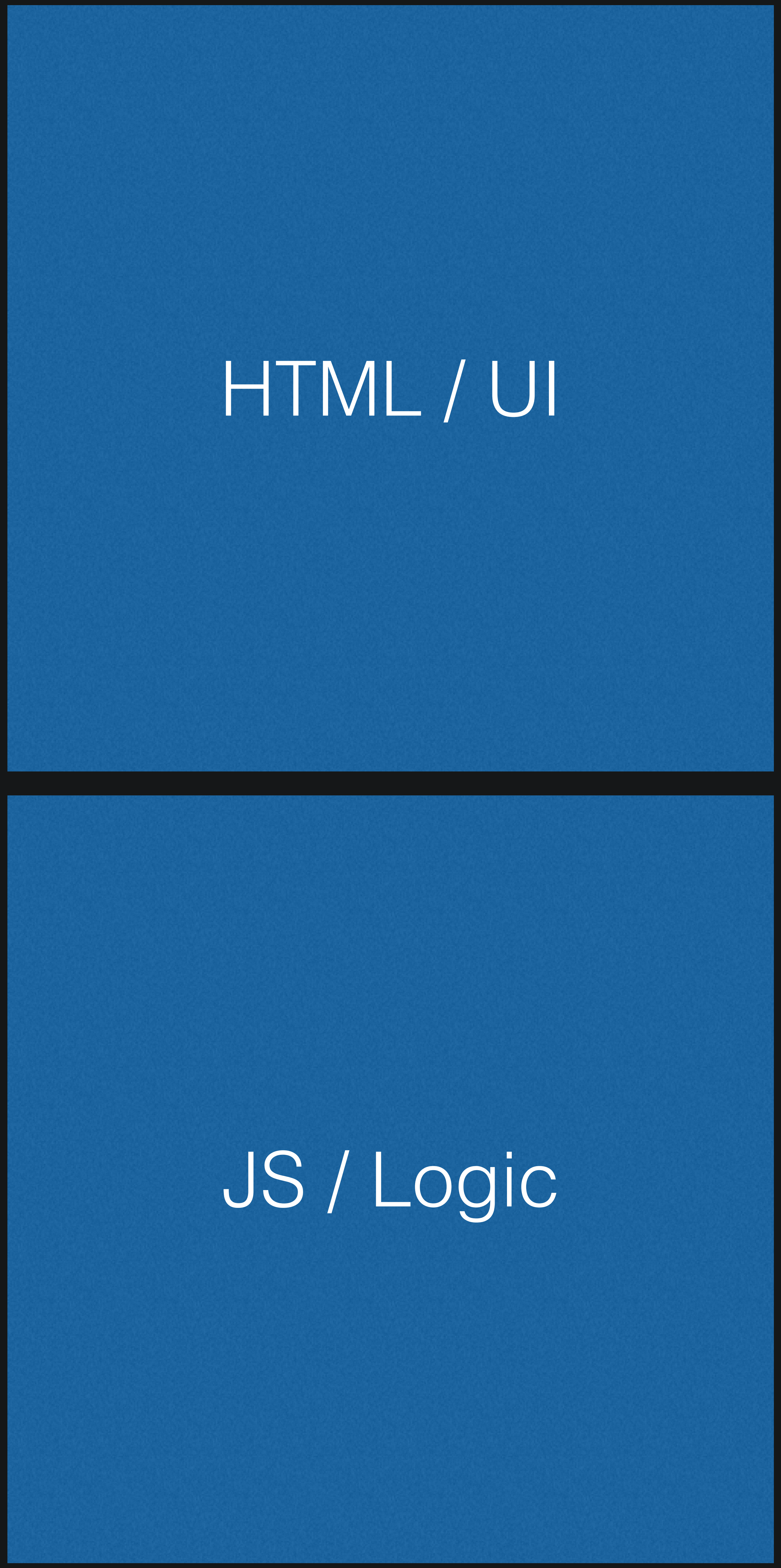
- Removing div soup.
- Performance.
- Less ID collisions.
- "Semantic"
- Encapsulated JS & CSS



```
<deadlift></deadlift>
```

WebWorker

- "Threads"
- JavaScript won’t block UI thread



Native

- Declarative UI means no need for HTML
- Supports web components
- … and native components

Goals

- Improve on Version 1
- Embrace Web Components
- Embrace Web Standards

Languages

You have options:

- TypeScript
- Dart
- ES6
- ES5

TypeScript

- Microsoft’s ES6
- opt-in strong-typing via compiler
- interop with other JavaScript via d.ts

Why TypeScript?

- strong-typing for larger code bases & teams
- runtime support via Assert.js
- a lot of syntax sugar
- invented by Anders Hejlsberg (Turbo Pascal, Delphi, and C#)
- output is readable

```typescript
class Greeter
{
    greeting: string;

    constructor(message: string)
    {
        this.greeting = message;
    }

    greet()
    {
        return "Hello, " + this.greeting;
    }
}

var greeter = new Greeter("world");
```

```javascript
var Greeter = (function () {
    function Greeter(message) {
        this.greeting = message;
    }
    Greeter.prototype.greet = function () {
        return "Hello, " + this.greeting;
    };
    return Greeter;
})();
var greeter = new Greeter("world");
```

Why not?

- … it’s not JavaScript, it’s TypeScript.
- Bet on JavaScript.

Dart

- Platform for the web & server by Google
- Use ES7, today, on client and server
- Opportunity to make a better web language

Why Dart?

- pub: better npm (… and so was BetaMax, and HDVideo)
- Dart: better Javascript
- DartVM: optimized for language
- dart2js: tree shaking, smart compiler, optimized JS

```dart
import 'dart:html' show HttpRequest;

main() async {
  // Asychronously get text to display.
  var lines = await getLines();

  // If result is non-null, print it.
  lines?.forEach((line) => print(line));
}

// Reads a file, returning all lines with the string
// 'jabberwock'.
getLines() async {
  var jabber = await HttpRequest.getString(
      'https://www.dartlang.org/f/jabberwocky.txt');
  var lines = jabber.split('\n');
  lines.retainWhere((line) =>
      line.toLowerCase().contains('jabberwock'));
  return lines;
}
```

Why not?

- Interop is not as easy as TypeScript
- Node.js adverse
- Convert npm to pub? That’s not happening…

ES6

- Next version of JavaScript
- Many parts implemented today
- Many already have polyfills

Why?

- The Standard.
- OOP: Classes.
- Modules: Standards coming finally.
- All features work in all other languages mostly.
- Node’s going there too.

```
class SkinnedMesh extends THREE.Mesh
{
    constructor(geometry, materials)
    {
        super(geometry, materials);

        this.idMatrix = SkinnedMesh.defaultMatrix();
        this.bones = [];
        this.boneMatrices = [];
    }

    update(camera)
    {
        super.update();
    }

    static defaultMatrix()
    {
        return new THREE.Matrix4();
    }
}
```

Why not?

- Design by committee blows
- Business innovation is faster than standards boards
- What good is power if you don’t use it?

ES5

- Also known as JavaScript.

Why?

- Transpiler adverse.
- Team skill set.
- Company that embraced ES5.

Why not?

- Newer, better languages & tooling out there.
- … including JavaScript. ES6.
- If you want functional, use ClojureScript.

Module System?

- No standard yet. You can use any for now.
- Browserify
- SystemJS
- WebPack
- JSPM

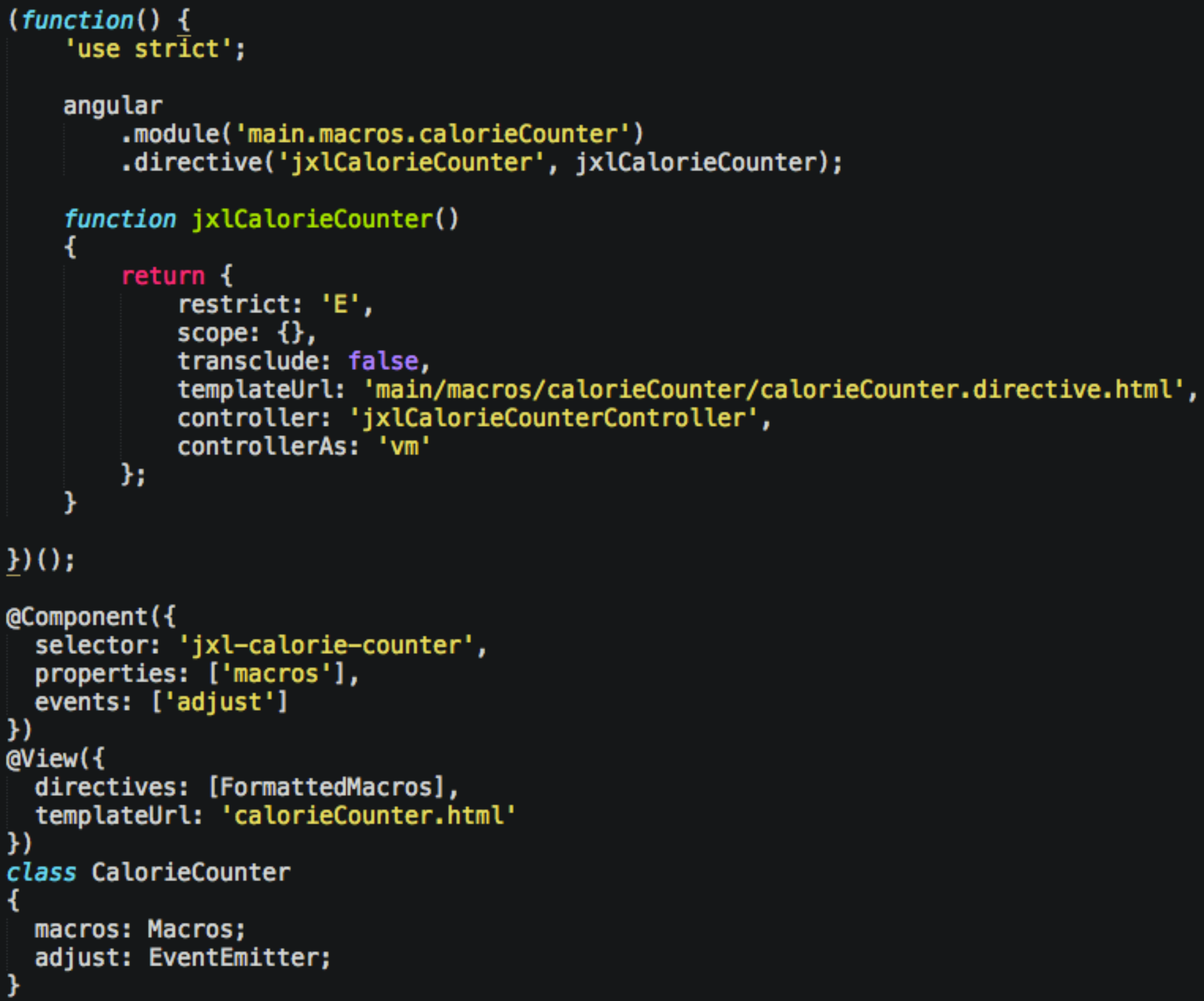
What’s Changed?

Angular 1 vs Angular 2 Alpha

Components

- Directives are now Components

Angular 1 vs 2 Example


```
(function() {
    'use strict';

    angular
        .module('main.macros.calorieCounter')
        .directive('jxlCalorieCounter', jxlCalorieCounter);

    function jxlCalorieCounter()
    {
        return {
            restrict: 'E',
            scope: {},
            transclude: false,
            templateUrl: 'main/macros/calorieCounter/calorieCounter.directive.html',
            controller: 'jxlCalorieCounterController',
            controllerAs: 'vm'
        };
    }

})();

@Component({
  selector: 'jxl-calorie-counter',
  properties: ['macros'],
  events: ['adjust']
})
@View({
  directives: [FormattedMacros],
  templateUrl: 'calorieCounter.html'
})
class CalorieCounter
{
  macros: Macros;
  adjust: EventEmitter;
}
```

React Example

```
var SetIntervalMixin = {
    componentWillMount: function() {
        this.intervals = [];
    },
    setInterval: function() {
        this.intervals.push(setInterval.apply(null, arguments));
    },
    componentWillUnmount: function() {
        this.intervals.forEach(clearInterval);
    }
};

var TickTock = React.createClass({
  mixins: [SetIntervalMixin], // Use the mixin
  getInitialState: function() {
    return {seconds: 0};
  },
  componentDidMount: function() {
    this.setInterval(this.tick, 1000); // Call a method on the mixin
},
tick: function() {
    this.setState({seconds: this.state.seconds + 1});
},
render: function() {
    return (
        <p>
        React has been running for {this.state.seconds} seconds.
        </p>
        );
}
});

ReactDOM.render(
    <TickTock />,
    document.getElementById('example')
    );
```

Polymer component

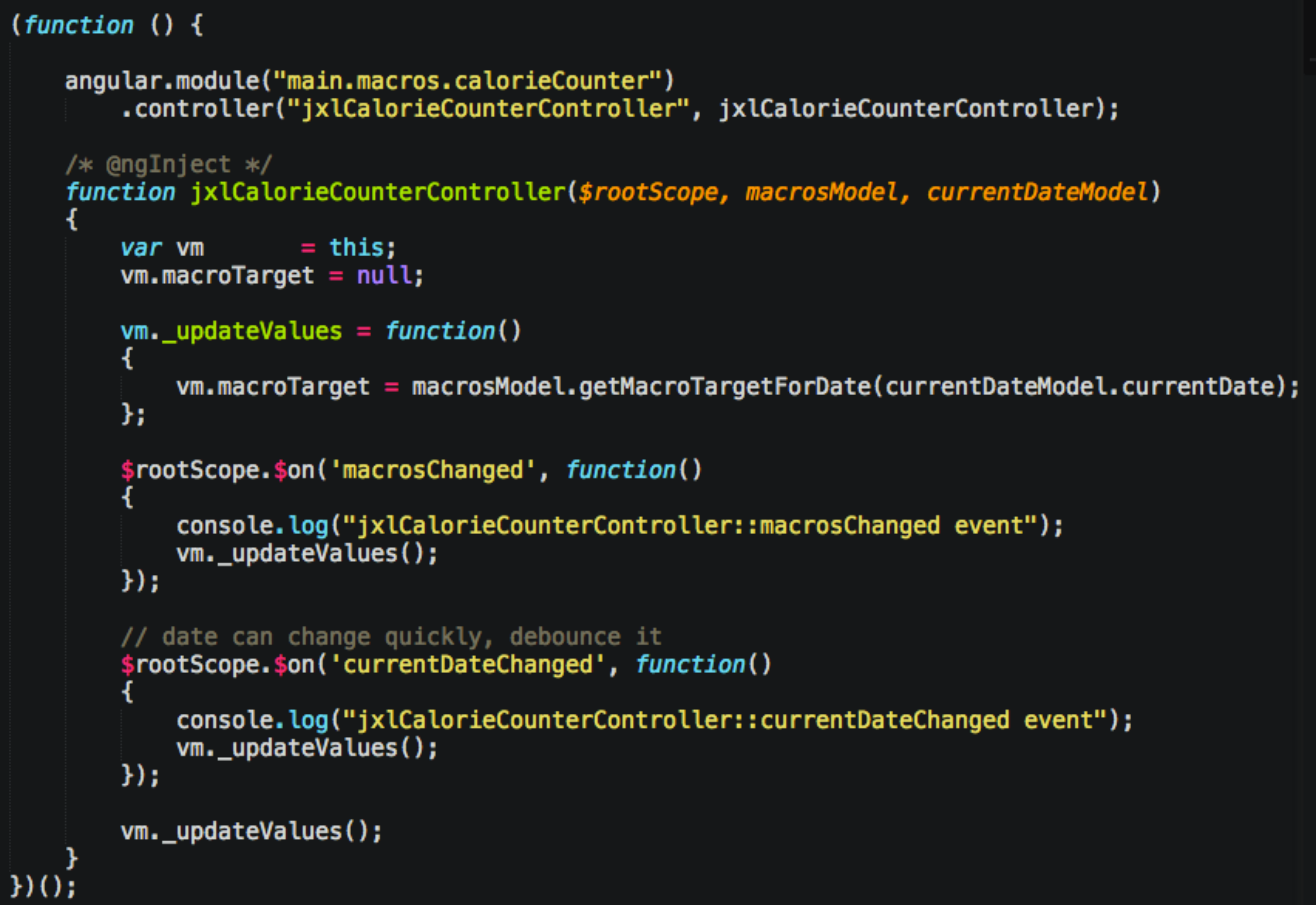
```
Polymer({
    is: "example-component",
    ready: function() {
        this.textContent = "Example element."
    }
});
```

```
<link rel="import" href="example-component.html">

<example-component></proto-component>
```

Class

- Controllers are now a Class

Angular 1 Controller vs 2



```
(function () {

    angular.module("main.macros.calorieCounter")
        .controller("jxlCalorieCounterController", jxlCalorieCounterController);

    /* @ngInject */
    function jxlCalorieCounterController($rootScope, macrosModel, currentDateModel)
    {
        var vm       = this;
        vm.macroTarget = null;

        vm._updateValues = function()
        {
            vm.macroTarget = macrosModel.getMacroTargetForDate(currentDateModel.currentDate);
        };

        $rootScope.$on('macrosChanged', function()
        {
            console.log("jxlCalorieCounterController::macrosChanged event");
            vm._updateValues();
        });

        // date can change quickly, debounce it
        $rootScope.$on('currentDateChanged', function()
        {
            console.log("jxlCalorieCounterController::currentDateChanged event");
            vm._updateValues();
        });

        vm._updateValues();
    }
})();
```

Properties Lifecycle

```
@Component({
    selector: 'calorie-counter',
    properties: ['macros', 'calories'],
    lifecycle: [onChange]
})
class CalorieCounter
{
    macros;
    calories;
    onChange(changes)
    {
        //...
      }
    }
}
```

Bindings to Inject

```
@Component({
    selector: 'fitness-app',
    bindings: [NgIf, Macros]
})
class FitnessApp
{
    //...
}

class CalorieCounter
{
    constructor(ngIf:NgIf, macros:Macros)
    {
        //...
    }
}
```

Host Element

```
@Component({
    selector: 'calorie-counter',
    host: {
        '(calorieNumericStepper)': 'onChange($event.target.value)',
        '[calories]': 'calories'
    }
    })
class CalorieCounter
{
    calories: number;
    onChange(updatedValue: string)
    {
        this.value = ensureValueIsNumberAndNotNaN(updatedValue);
    }
}
```

Templates

- Templates are now Views

v1 Template vs v2

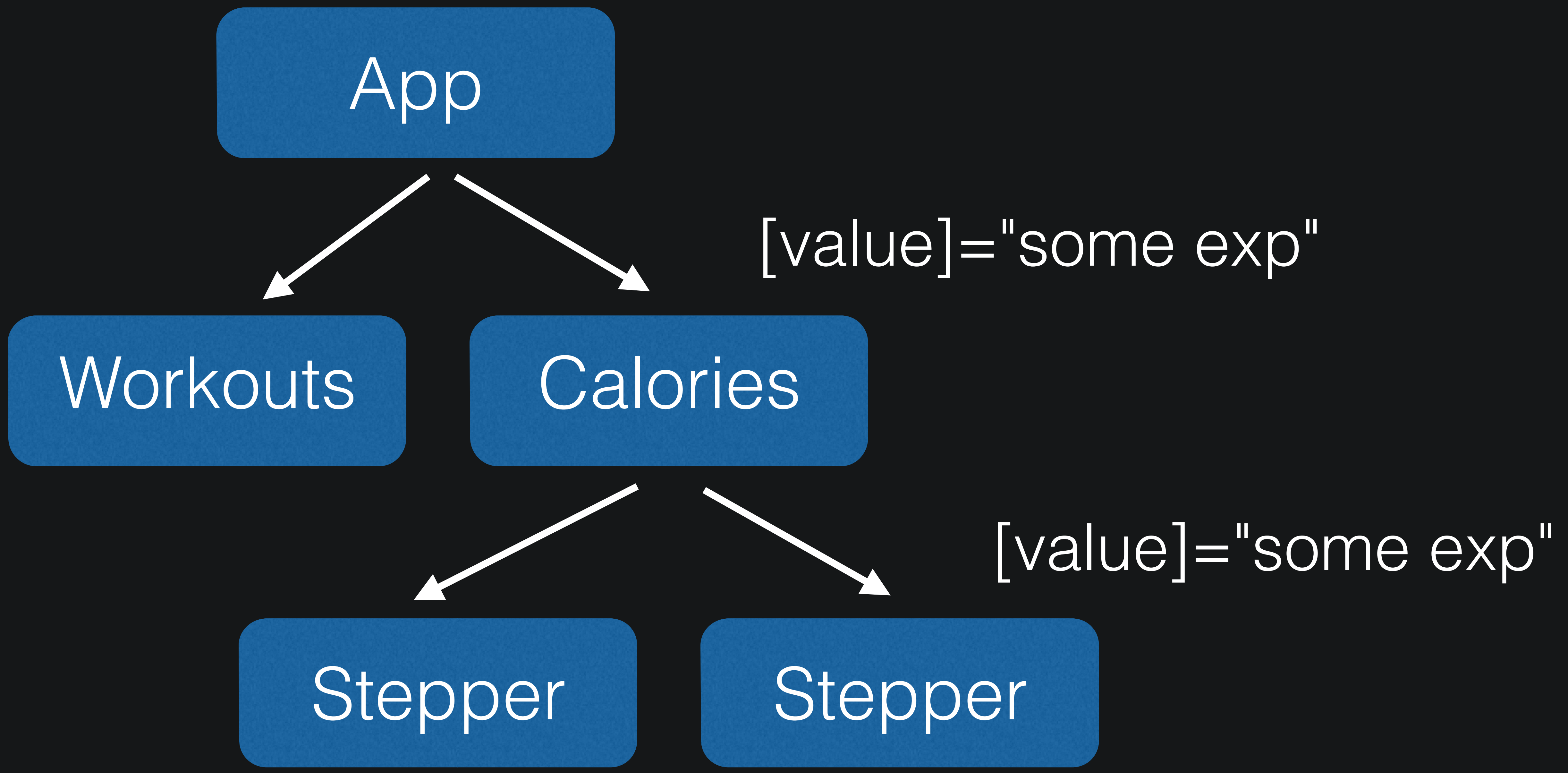
```html
<div class="row">
    <h2>Calories Remaining</h2>
    <p>{{vm.macroTarget.remaining}}</p>
    <div>
      <span style="margin: 0.75em;">{{vm.macroTarget.goal | number:0}}</span>
    </div>
    <div>
      <span style="margin: 0.75em;"><small>Goal</small></span>
    </div>
    <stepper value="{{vm.macroTarget.calories}}" ng-model="calories"></stepper>
</div>

<div class="row">
    <h2>Calories Remaining</h2>
    <p>{{macroTarget.remaining}}</p>
    <div>
      <formatted-goal [goal]="goal"></formatted-goal>
    </div>
    <div>
      <span style="margin: 0.75em;"><small>Goal</small></span>
    </div>
    <stepper [value]="calories" (change)="onChange($event.target.value)"></stepper>
</div>
```

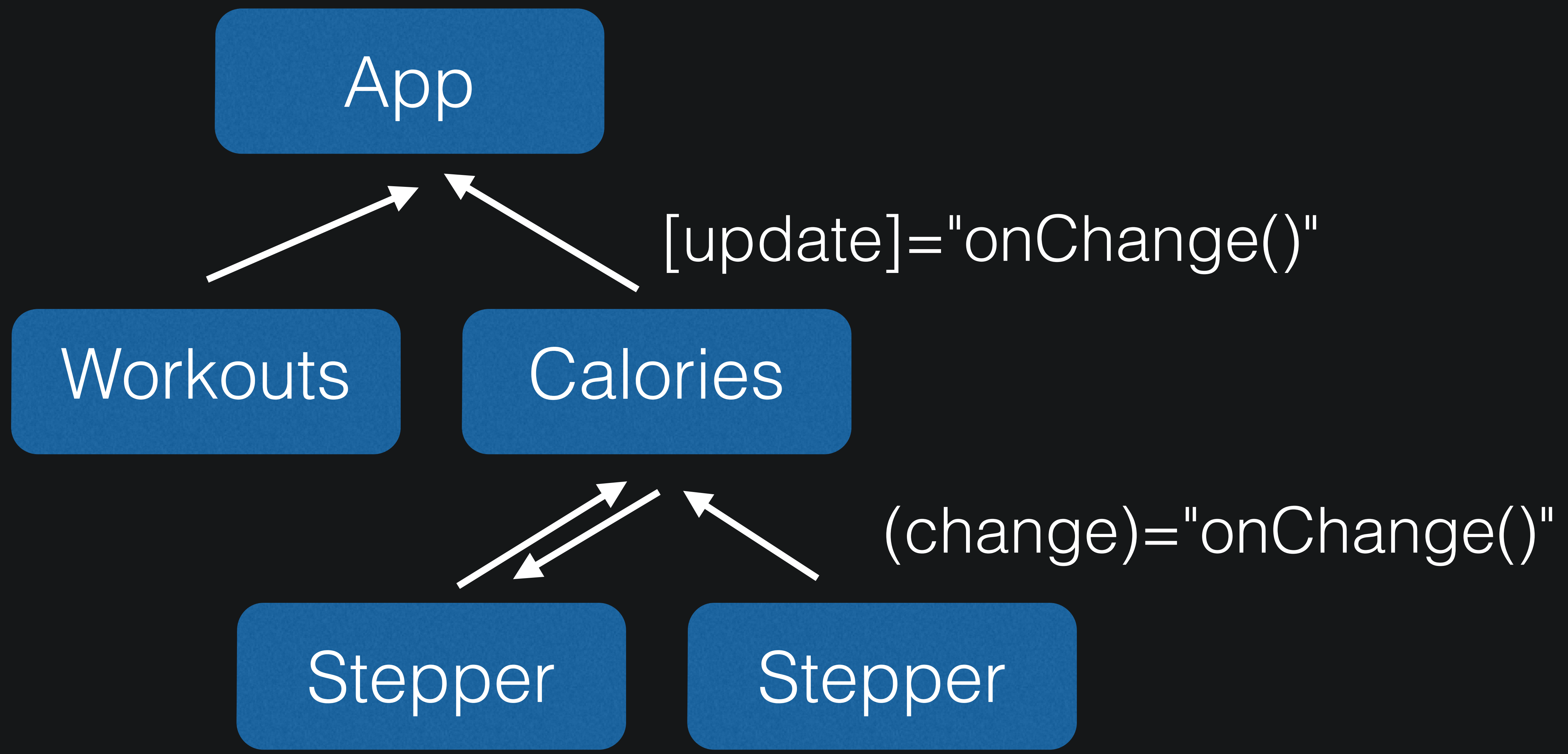
2 way binding

```html
<stepper [value]="calories" (change)="onChange($event.target.value)"></stepper>

<stepper [(value)]="calories" (change)="onChange($event.target.value)"></stepper>
```

Parent -> Child

Child -> Parent





ng-repeat vs *ng-for

```
<workouts *ng-for="#w of workouts; #i=index" [model]="w" [index]="i"></workouts>

<workouts template="ng-for #w of workouts; #i=index" [model]="w" [index]="i"></todo-cmp>

<template ng-for [ng-for-of]="workouts" #w="$implicit" #index="i">
    <workouts [model]="w" [index]="i"></workouts>
</template>
```

Local Variables

```
<deadlift #workout></deadlift>
<button (click)="workout.addSet()">Add Set</button>
```

overview of digest vs. tree invalidation

bindings treated as immutable

2 digest loops; if you set data in 2nd zone, it’ll throw an exception letting you know you’ve got an infinite loop

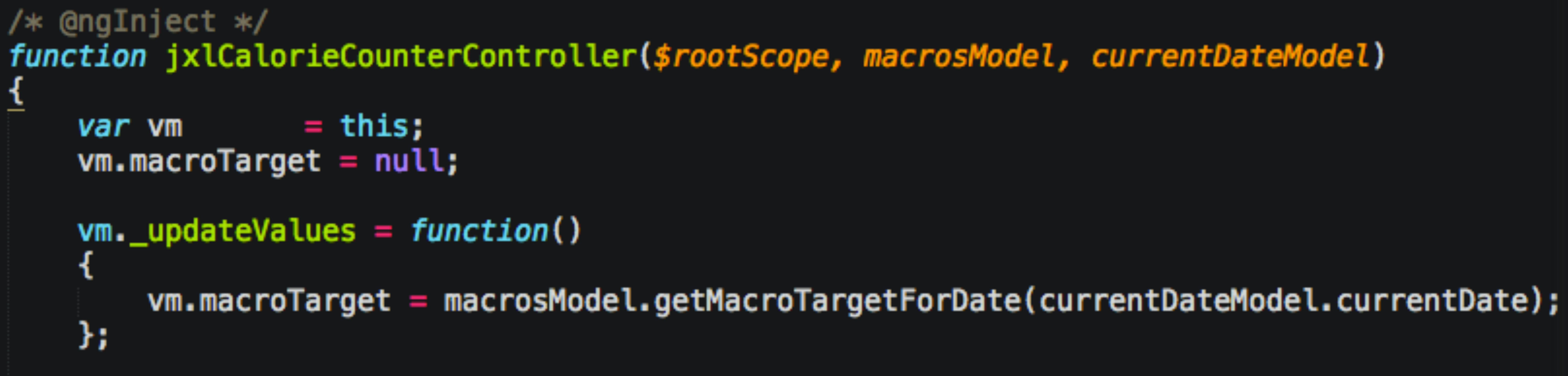
Zone.js

Directives

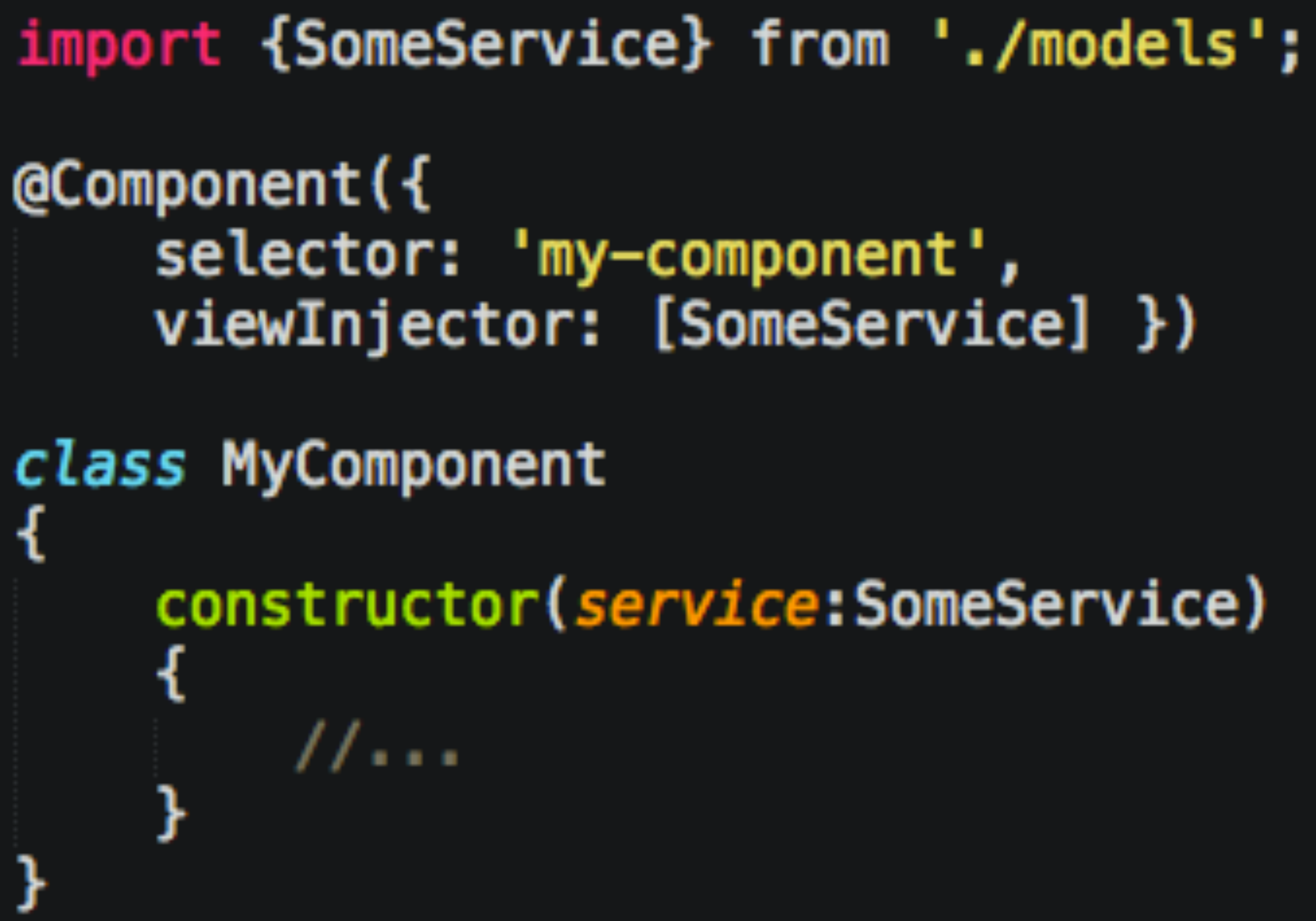
- Can still create directives that don’t have a View

Services, Factories, etc

- Services, Factories, Filters, etc. are now classes
- You still inject via DI
- if not using classes, still have new DI functionality



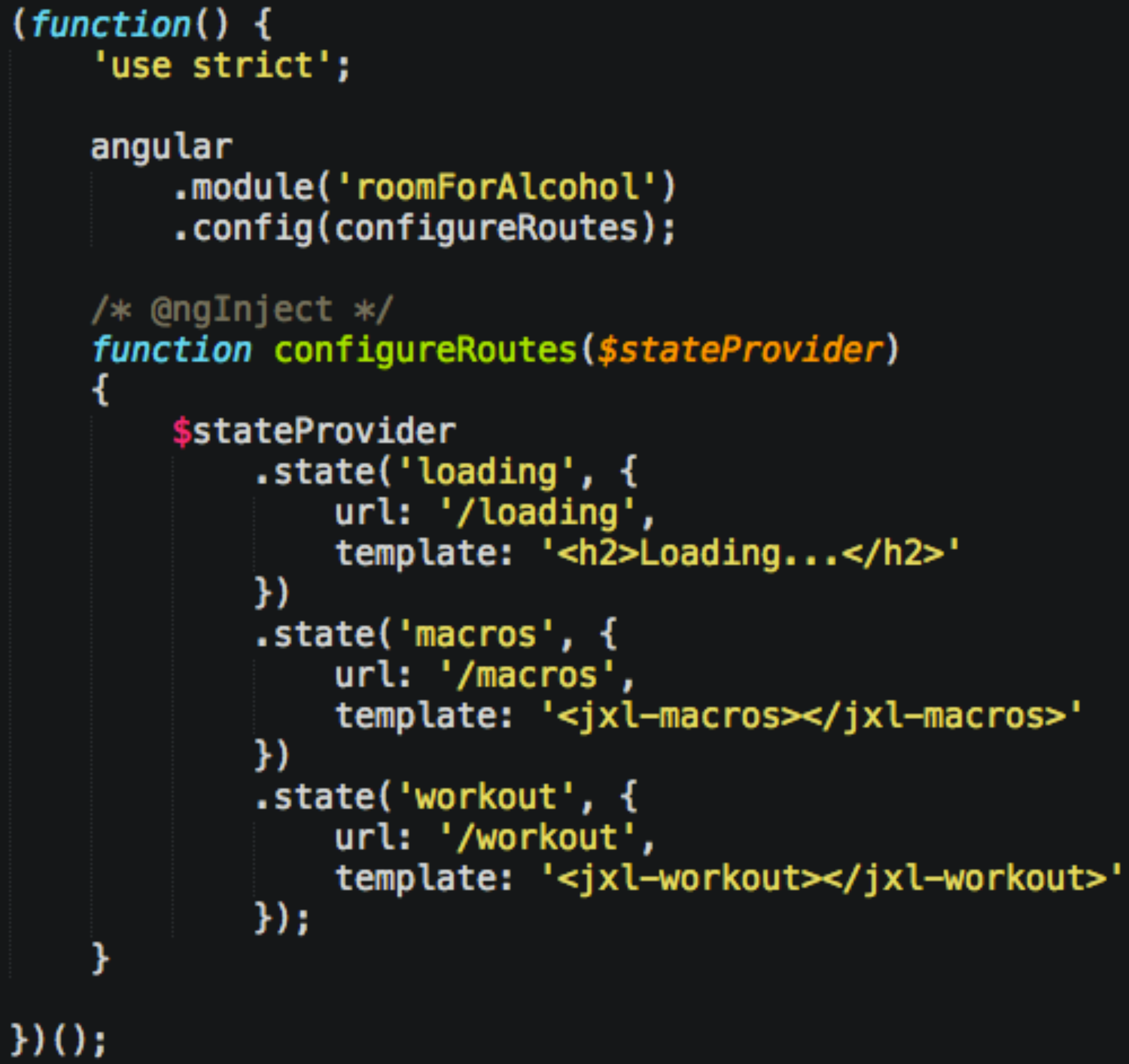
```
/* @ngInject */
function jxlCalorieCounterController($rootScope, macrosModel, currentDateModel)
{
    var vm         = this;
    vm.macroTarget = null;

    vm._updateValues = function()
    {
        vm.macroTarget = macrosModel.getMacroTargetForDate(currentDateModel.currentDate);
    };
```



```
import {SomeService} from './models';

@Component({
    selector: 'my-component',
    viewInjector: [SomeService] })

class MyComponent
{
    constructor(service:SomeService)
    {
        //...
    }
}
```

New Router

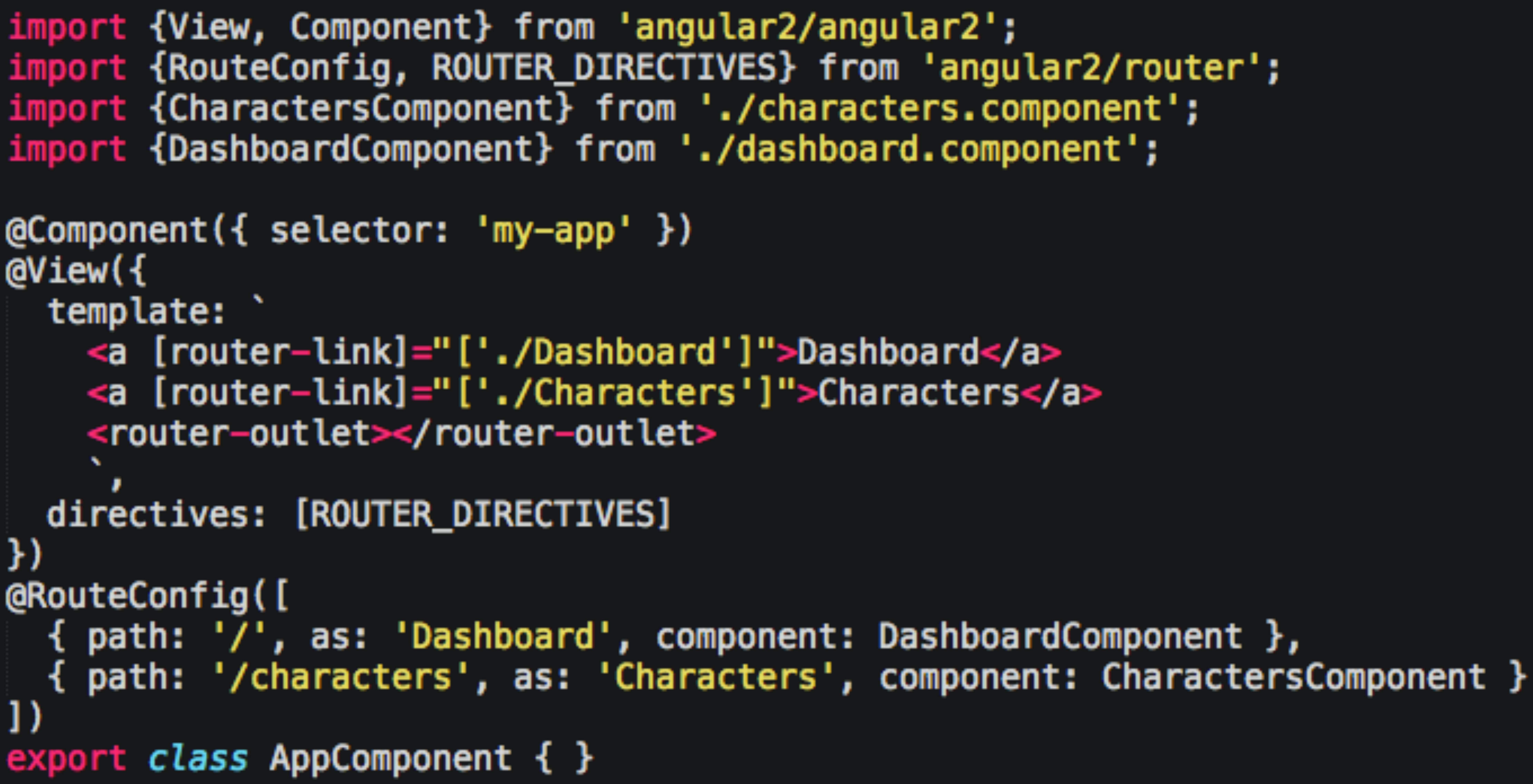
- ngRoute is now ‘New Router’
- same one they introduced in 1.4



```
(function() {
    'use strict';

    angular
        .module('roomForAlcohol')
        .config(configureRoutes);

    /* @ngInject */
    function configureRoutes($stateProvider)
    {
        $stateProvider
            .state('loading', {
                url: '/loading',
                template: '<h2>Loading...</h2>'
            })
            .state('macros', {
                url: '/macros',
                template: '<jxl-macros></jxl-macros>'
            })
            .state('workout', {
                url: '/workout',
                template: '<jxl-workout></jxl-workout>'
            });
    }

})();
```

John Papa’s Angular 2 Sample

https://github.com/johnpapa/angular2-go



```
import {View, Component} from 'angular2/angular2';
import {RouteConfig, ROUTER_DIRECTIVES} from 'angular2/router';
import {CharactersComponent} from './characters.component';
import {DashboardComponent} from './dashboard.component';

@Component({ selector: 'my-app' })
@View({
  template: `
    <a [router-link]="['./Dashboard']">Dashboard</a>
    <a [router-link]="['./Characters']">Characters</a>
    <router-outlet></router-outlet>
    `,
  directives: [ROUTER_DIRECTIVES]
})
@RouteConfig([
  { path: '/', as: 'Dashboard', component: DashboardComponent },
  { path: '/characters', as: 'Characters', component: CharactersComponent }
])
export class AppComponent { }
```

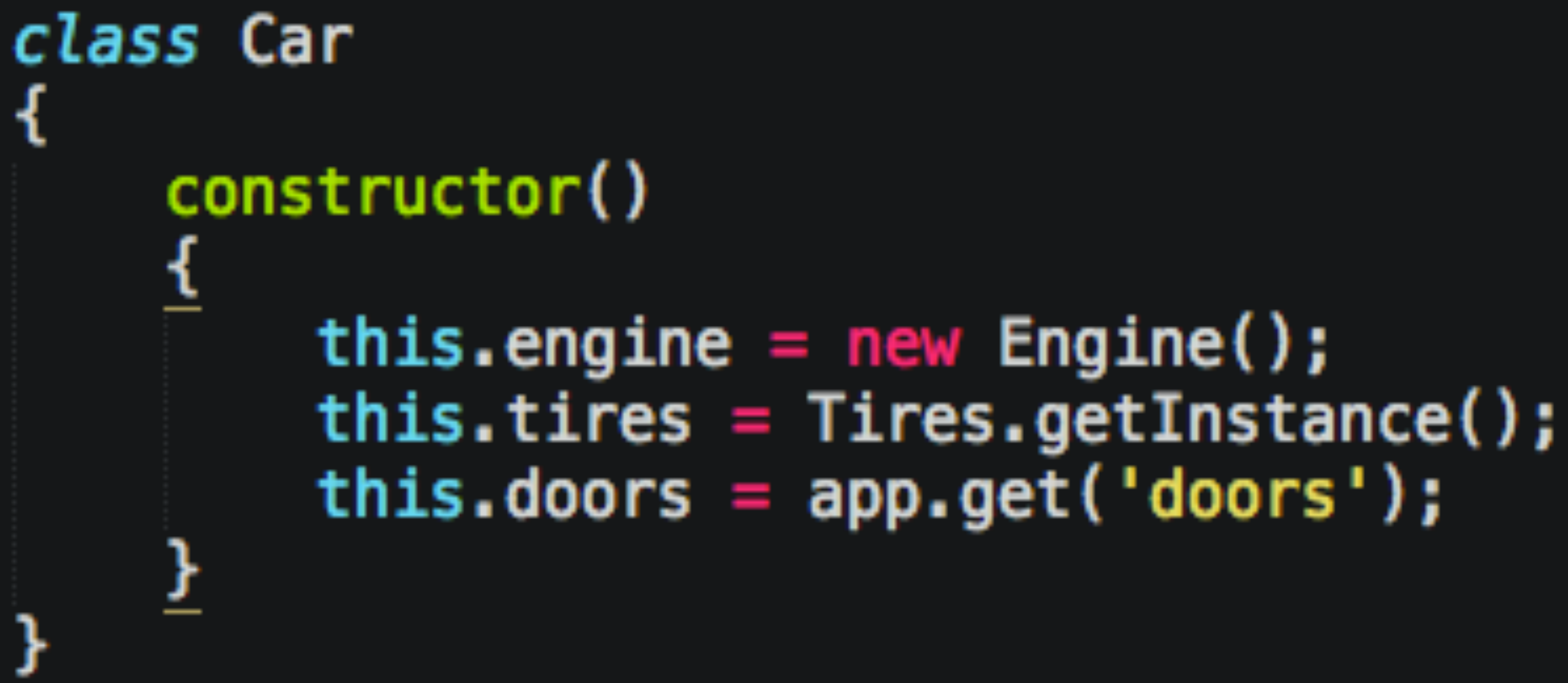
HTTP

- $http is now HTTP class with upgraded functionality
- slides on streams: https://docs.google.com/file/d/0B8xUu4uAO8rnbVBkd0l6M285aFk/edit
- Sample code for streams: https://gist.github.com/JesterXL/d2f89ccb17b26574b233

```
http.get('http://server.com/nastyJavaSoap.xml')
.toRx()
.map(response => response.json())
.subscribe(result => this.redrawFromData(result));
```

Dependency Injection

- new functionality!!!!1111oneonene

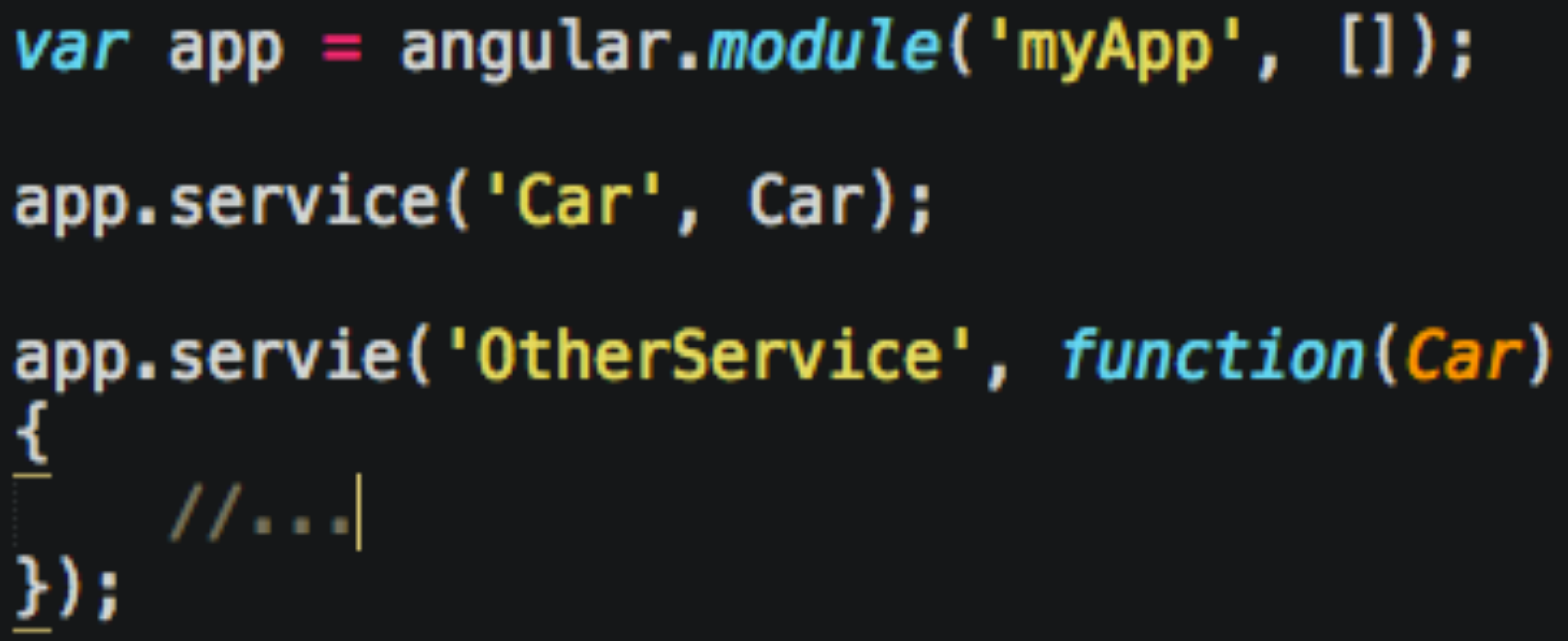


```
class Car
{
    constructor()
    {
        this.engine = new Engine();
        this.tires = Tires.getInstance();
        this.doors = app.get('doors');
    }
}
```

```
class Car
{
    constructor(engine, tires, doors)
    {
        this.engine = engine;
        this.tires = tires;
        this.doors = doors;
    }
}
```

```
var car = new Car(
    new Engine(),
    new Tires(),
    new Doors()
);
```

```
var car = new Car(
    new MockEngine(),
    new MockTires(),
    new MockDoors()
);
```



```
var app = angular.module('myApp', []);

app.service('Car', Car);

app.servie('OtherService', function(Car)
{
    //...
});
```

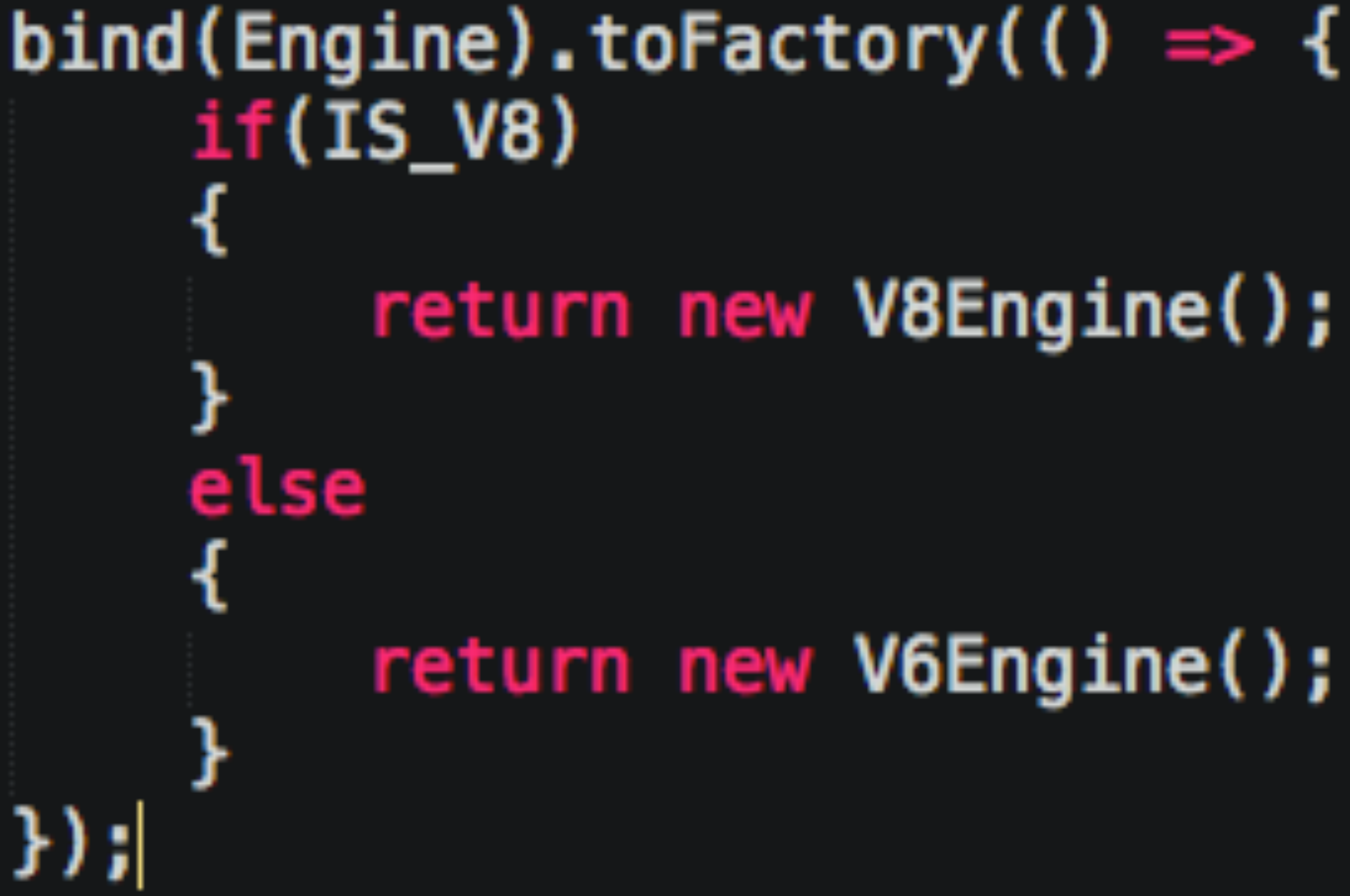
```
import {Injector} from 'angular2/di';

var injector = Injector.resolveAndCreate([
    Car,
    Engine,
    Tires,
    Doors
]);

var car = injector.get(Car);
```

```
import {Injector} from 'angular2/di';

class Car
{
    constructor(
        @Inject(Engine) engine,
        @Inject(Tires) tires,
        @Inject(Doors) doors
        )
    {
        //...
    }
}
```

```
import {Inject} from 'angular2/di';

class Car
{
    constructor(engine:Engine, tires:Tires, doors:Doors)
    {
        //...
    }
}
```

```
import {bind} from 'angular2/di';

var injector = injector.resolveAndCreate([
    bind(Car).toClass(Car),
    bind(Engine).toClass(Engine),
    bind(Tires).toClass(Tires),
    bind(Tires).toClass(Doors)
]);
```

```
bind(Engine).toClass(OtherEngine)

bind(Engine).toClass(MockEngine)
```



```
bind(Engine).toFactory(() => {
    if(IS_V8)
    {
        return new V8Engine();
    }
    else
    {
        return new V6Engine();
    }
});
```

Conclusions

- Angular 2 is Alpha, not for production
- API still changing
- but you can now play with Developer Preview
- TypeScript, Dart, ES6, and/or ES5
- ES5 examples are lagging, but coming

Resources

- Angular 2 site: https://angular.io/
- John Papa simple repo: https://github.com/johnpapa/angular2-go
- Thomas Manion WebPack repo: https://github.com/1337programming/angular2.0-Wepack-App
- Victor Savkin: http://victorsavkin.com/
- http://blog.thoughtram.io/
- TypeScript: http://www.typescriptlang.org/Handbook
- ES6: https://github.com/lukehoban/es6features

Thanks!

- Jesse Warden
- jesse.warden@accenture.com
- jesse.warden@gmail.com
- @jesterxl
- http://jessewarden.com/blog/
- https://www.youtube.com/user/jesterxl