ANGULAR

André Werlang

@awerlang

The Developer's Conference 2017 - Porto Alegre

TODAY'S MISSION

- Remove barrier to TypeScript
- Get an app running is easy
- Understand how it works

AGENDA

- Myths & Facts
- Getting Started
- Features
- Elements
- Final Remarks

ANGULAR 2 MYTHS

- Requires TypeScript TRUTH
- Doesn't support two-way binding MYTH
- Requires a master's degree on bundling tools MYTH
- View code is VM friendly TRUTH
- Has its own module loading system MYTH
- Run side by side with Angular 1.x TRUTH

FACTS

- ~30k stars on GitHub
- ~7.5k forks on GitHub
- ~2.5k watchers on GitHub
- > 80k questions on StackOverflow
- > 14k users on gitter

GETTING STARTED

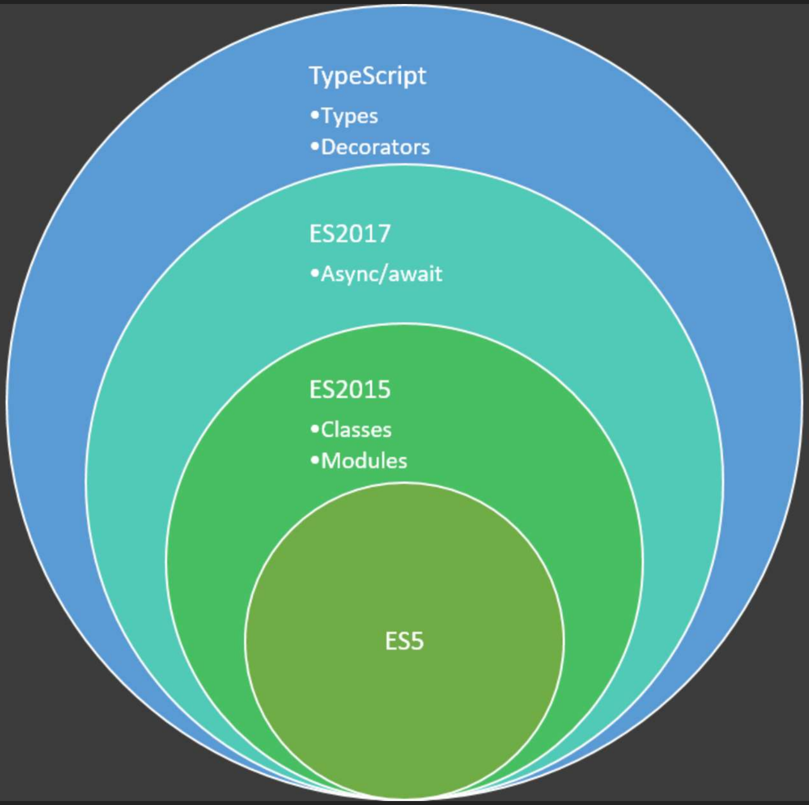
POWERED BY

JavaScript

TypeScript

RxJS

TYPESCRIPT



https://www.typescriptlang.org/

COMPILER (TSC)

- Targets ES5, ES2015, ES2016...
- Module generation: ES2015, CommonJS, System...
- Emits decorator metadata
- Many static checks
- Watch mode

LANGUAGE

- Type annotations
- Interfaces
- Enums
- Visibility modifiers
- Auto-properties on constructors
- Structural compatibility

DATA TYPES

- any
- string
- number
- boolean
- null
- undefined
- void
- never

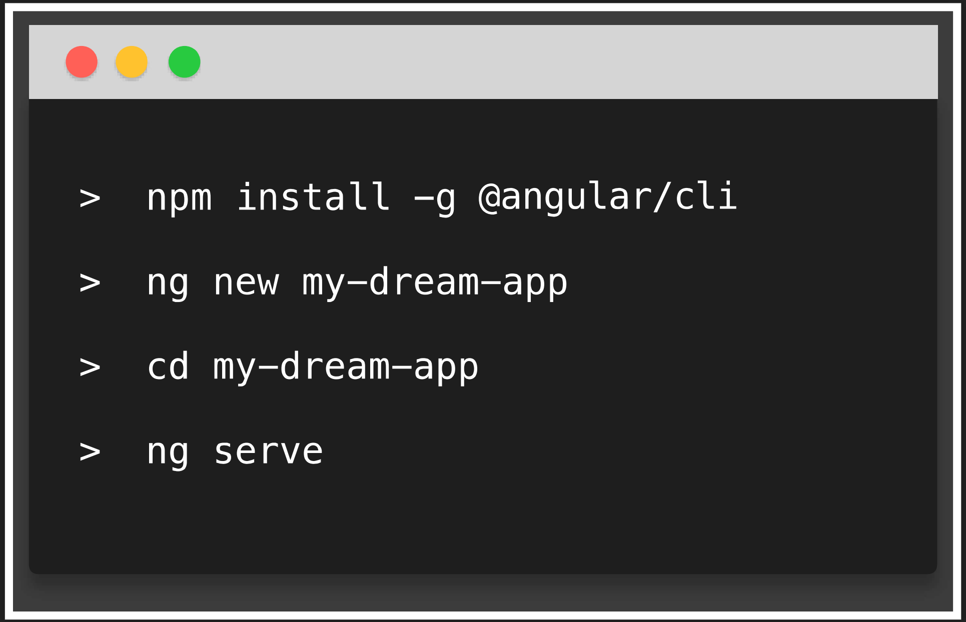
MIGRATE FROM JS

- Declare external definitions
- Add types to every declaration
- Create classes for implicit structures
- Convert constructor functions to classes
- Declare implicit members
- (Optional) Convert anonymous functions to arrow syntax
- (Optional) Use block-scoped bindings (const, let)

https://www.typescriptlang.org/docs/handbook/migrating-from-javascript.html

ANGULAR CLI

A command line interface for Angular projects



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

>  ng new my-dream-app

>  cd my-dream-app

>  ng serve
```

https://github.com/angular/angular-cli

ANGULAR CLI

COMMANDS

Create component

```
ng g c <component-name>
```

Serve

```
ng serve
```

Build production, offline compiler

```
ng build --prod --aot
```

Linter

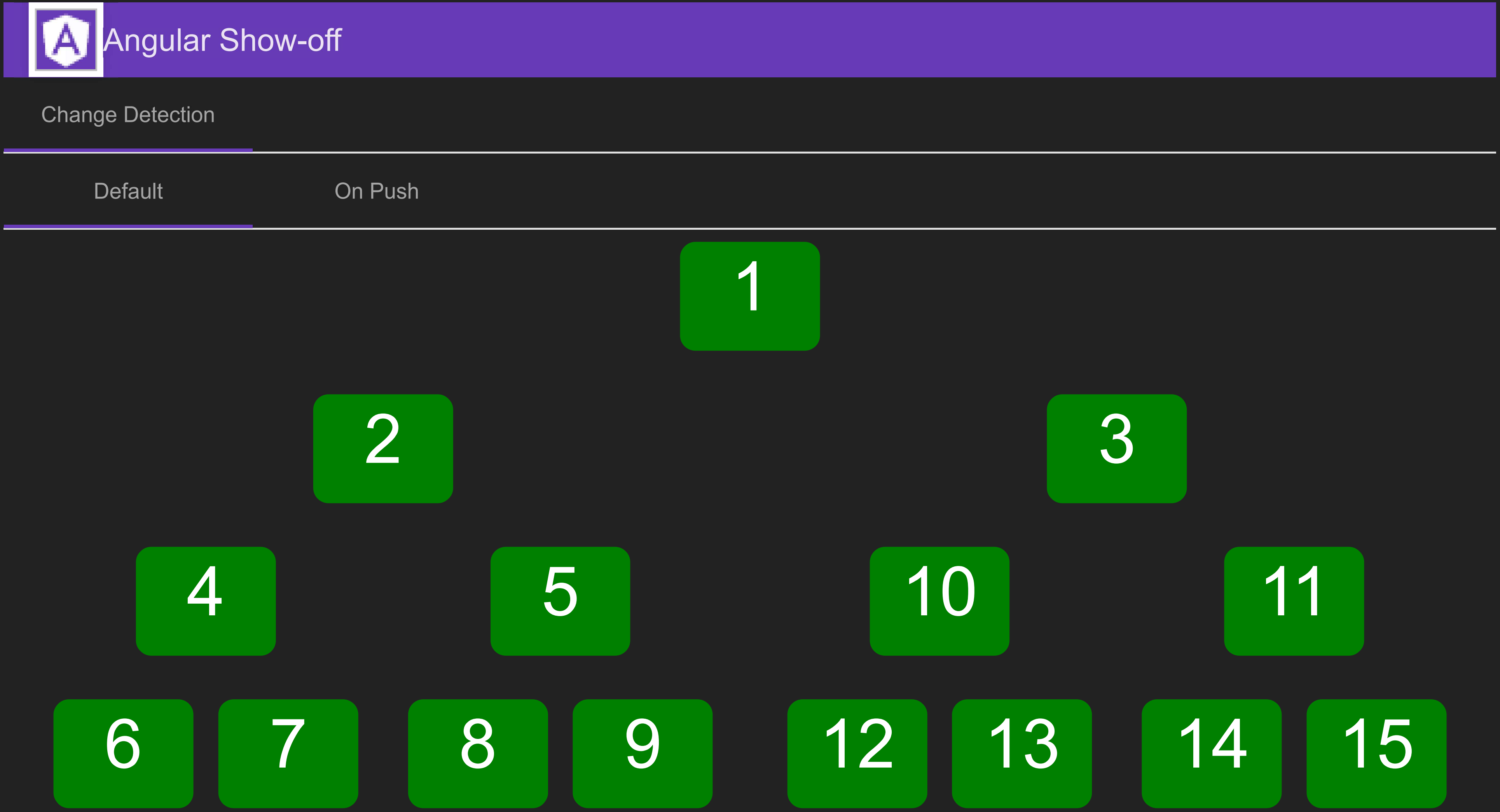
```
ng lint
```

Test

```
ng test
```

FEATURES

CHANGE DETECTION

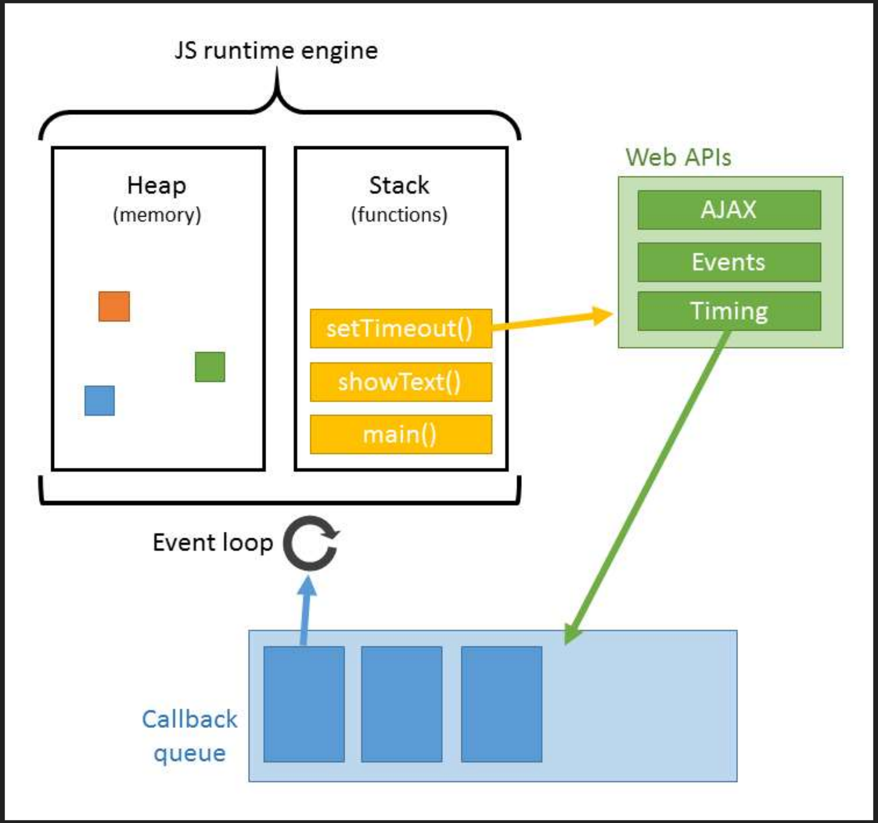


It's interactive! Click on any element!

TAKEAWAYS

- Change detection happens top-down
- Needs to stabilize in a single round
- Shared, mutable structures is a no-go
- On Push performs better (and is not that hard!)

ZONES



https://domenic.github.io/zones/

ZONES

NO MORE:

- $q
- $timeout
- $scope.$apply (well, almost)
- $scope.$$phase!!

DECORATORS

They augment:

- Classes
- Properties
- Methods
- Parameters

```
@frozen class Foo {
  @configurable(false) @enumerable(true) method() {}
}
```

https://tc39.github.io/proposal-decorators/

https://www.typescriptlang.org/docs/handbook/decorators.htm

ELEMENTS

COMPONENTS

Provides a context for data and events,
supports template, styling,
can have services injected
and is change detected

* A directive is a component with no template.

COMPONENTS

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  topics = ['commits', 'branches', 'remotes'];
  actions = ['list', 'create', 'delete'];

  onGoClick() {
    // TODO
  }
}
```

COMPONENTS

INPUTS & OUTPUTS

```
import { Component, Input, Output, EventEmitter } from '@angular/core

@Component({
  selector: 'my-component',
  templateUrl: './my.component.html',
  styleUrls: ['./my.component.css']
})
export class MyComponent {
  @Input() input: DataType;

  @Output() event = new EventEmitter<DataType>(false);

  onEvent(value) {
      this.event.emit(value);
  }
}
```

TEMPLATES

Composes the view with an HTML-like syntax,
interpolates text and data, binds events,
and includes other components.

TEMPLATES

```
<md-chip-list>
  <md-chip *ngFor="let topic of topics">{{topic}}</md-chip>
</md-chip-list>

<button md-button color="primary" (click)="onGoClick()">Go!</button>
```

TEMPLATES

INPUTS & OUTPUTS

```
<input #myInput [value]="input" (change)="myInput.value">
```

```
<input [(ngModel)]="field">
```

STYLING

Styles a component view, supports view encapsulation.

STYLING

```css
:host {
    display: block;
    max-width: 500px;
    font-family: Roboto,"Helvetica Neue",sans-serif;
}

.go {
    margin-top: 8px;
    float: right;
}

md-divider {
    clear: both;
}
```

ANIMATIONS

Declarative transitions and animations,
with state triggers and synchornization support.

ANIMATIONS

```
animations: [
  trigger('flyInOut', [
    state('in', style({transform: 'translateX(0)'})),
    transition('void => *', [
      style({transform: 'translateX(-100%)'}),
      animate(100)
    ]),
    transition('* => void', [
      animate(100, style({transform: 'translateX(100%)'}))
    ])
  ])
]
```

```
<md-chip *ngFor="let topic of topics" [@flyInOut]="'in'">...</md-chip
```

SERVICES

Encapsulates business logic
and shares data among components

SERVICES

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

@Injectable()
export class TipsService {
    match(topic, action) {
        // TODO
    }
}
```

```
import { TipsService } from './tips.service';

export class AppComponent {
  constructor(private tips: TipsService) { }

  onGoClick() {
    const matchingTips = this.tips.match(this.topic, this.action);
  }
}
```

NGMODULES

Organizes elements and creates reusable modules.

NGMODULES

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { MaterialModule } from '@angular/material';
import { AppComponent } from './app.component';
import { TipsService } from './tips.service';

@NgModule({
  declarations: [AppComponent],
  imports: [
    BrowserModule,
    MaterialModule.forRoot(),
  ],
  providers: [TipsService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

TESTING

COMPONENTS

```
beforeEach(() => {
  TestBed.configureTestingModule({
    imports: [
      MaterialModule.forRoot(),
    ],
    declarations: [
      AppComponent
    ],
    providers: [{ provide: TipsService, useValue: tipsService }],
  });
  TestBed.compileComponents();
});
```

COMPONENTS

```
it('should render results', async(() => {
  const topicChip: DebugElement = fixture.debugElement
    .query(By.css('.topics')).query(By.directive(MdChip));
  const actionChip: DebugElement = fixture.debugElement
    .query(By.css('.actions')).query(By.directive(MdChip));

  topicChip.triggerEventHandler('click', null);
  actionChip.triggerEventHandler('click', null);

  goBtn.triggerEventHandler('click', null);
  fixture.detectChanges();

  expect(el.querySelector('h4').textContent).toEqual('One tip found')
}));
```

SERVICES

```
let service = new TipsService();
beforeEach(() => {
    service = new TipsService();
});

describe('getTopics', () => {
    it('should work', () => {
        expect(service.getTopics())
            .toEqual(['changes', 'branches', 'commits', 'remotes']);
    });
});
```

STANDARD MODULES

- Http
- Forms
- Router

FINAL REMARKS

PLATFORMS

- Progressive Web Apps
- Angular Universal
- Ionic
- NativeScript
- ReactNative

https://angular.io/resources/

UI COMPONENTS

- Material
- Bootstrap
- Lightning
- Semantic UI
- PrimeNG
- Kendo UI
- wijmo

https://angular.io/resources/

STATE CONTAINERS

- ngrx/store & friends
- ng2-redux

RECOMMENDATIONS

- Employ a redux architecture
- Use unidirectional data flow (OnPush strategy)
- Learn about containers & presentational components
- Turn on static checks for TypeScript

OBRIGADO!

@awerlang

https://github.com/awerlang/angular-show-off