

Angular 2 не так уж и плох,

а если задуматься то и просто хорош.

Алексей Охрименко (IPONWEB)



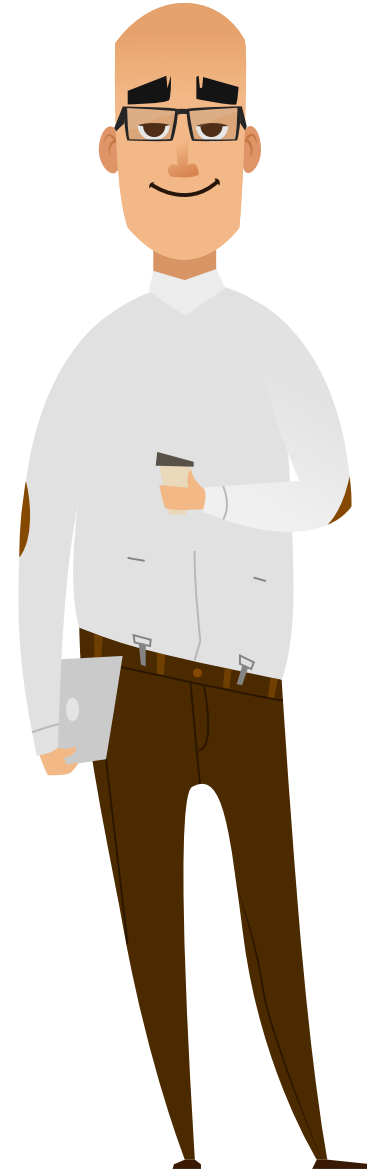
**Frontend
Conf**

Здоровье прежде всего

Алексей Охрименко

Tweeter: @Ai_boy

Gitter: aiboy



IPONWEB



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

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  template: `
    <h1 (click)='onClick()'>
      {{title}}
    </h1>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  title = 'project-name works!';
}
```

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

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  template: `
    <h1 (click)='onClick()'>
      {{title}}
    </h1>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  title = 'project-name works!';
}
```

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

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  template: `
    <h1 (click)='onClick()'>
      {{title}}
    </h1>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  title = 'project-name works!';
}
```

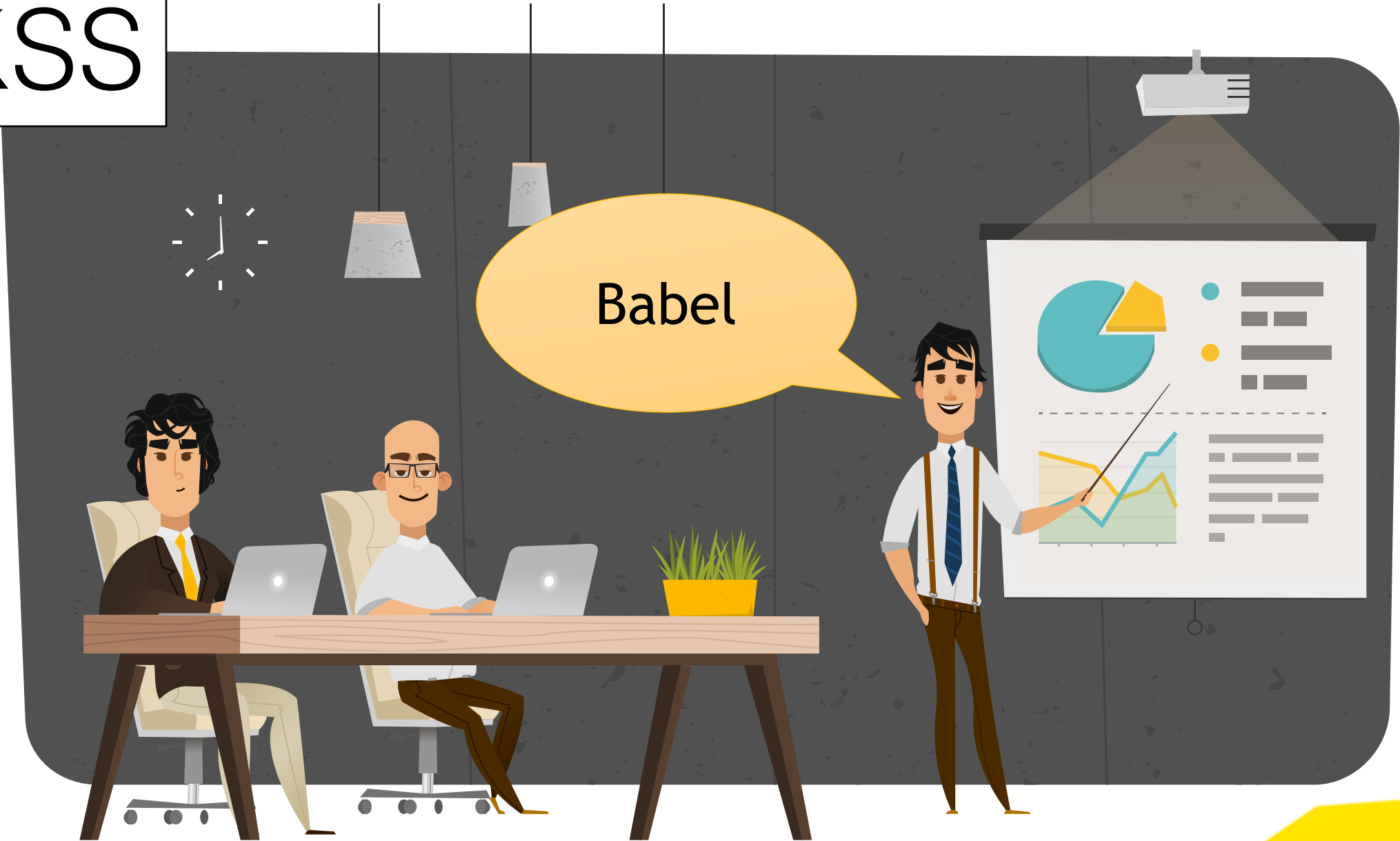
Глава №1 - И была рука



KSS



KSS



KSS

TypeScript?



KSS

Angular 2



Глава №2 - Горы отвращения



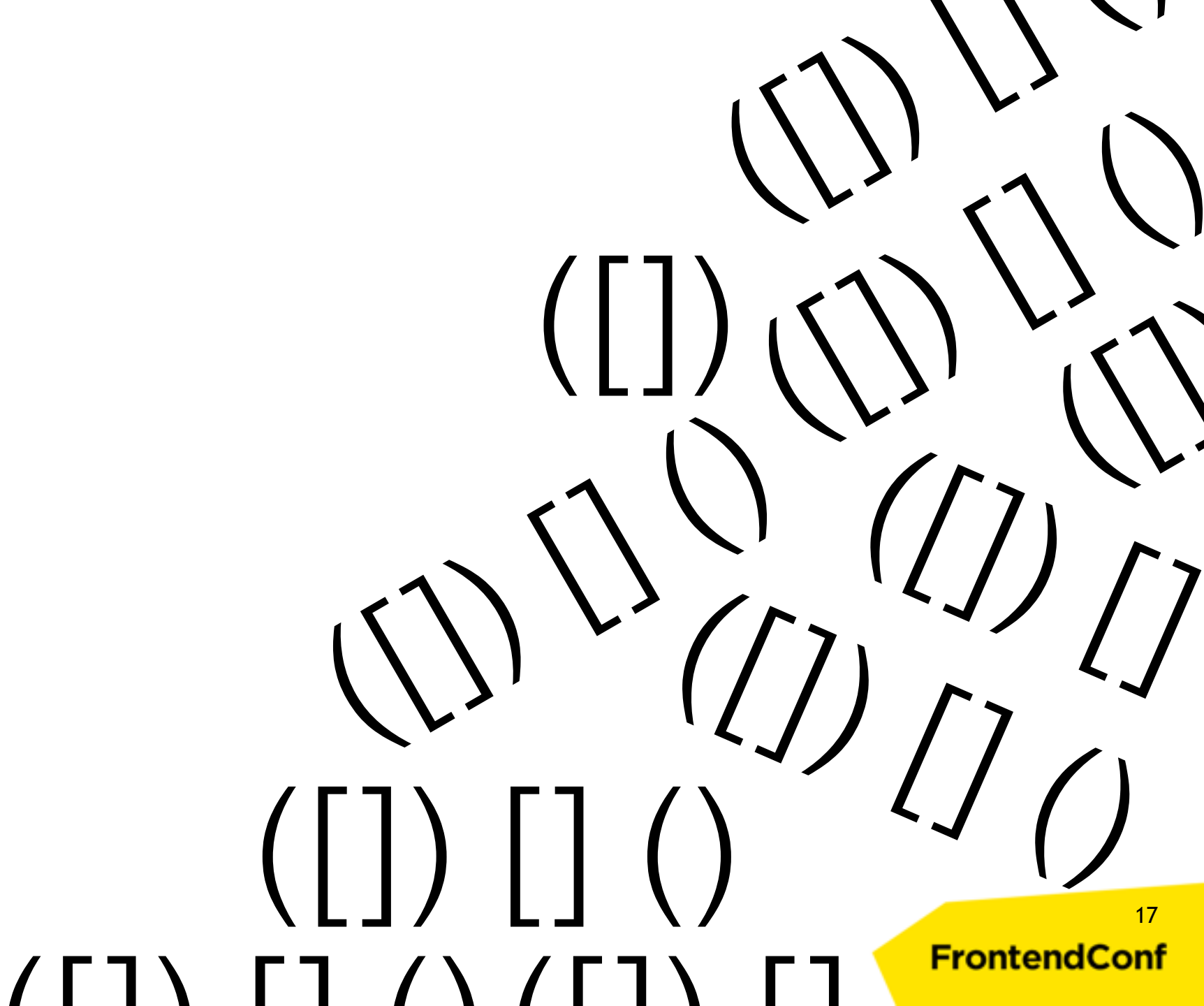
Angular 2 is Beta*

* - now RC1



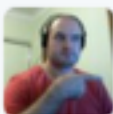
ASP.Net MVC - Developer Preview 2







You Retweeted



Jonny Buchanan @jbscript · 23 Nov 2015

Dear Angular 2. Wat.

Pin it

```
<ul>
  <li *ng-for="#user of users">
    {{ user.name }} is {{ user.age }} years old.
  </li>
</ul>
```

As you can see there's no more `ng-repeat`, it's `ng-for` NOW. You are probably thinking: "why the asterisk?!". The answer to that is, it's syntactic sugar. What you're really doing is:

```
<ul>
  <template ng-for #user [ng-for-of]="users">
    <li>{{ user.name }} is {{ user.age }} years old.</li>
  </template>
</ul>
```



Raymond Camden

@raymondcamden



Follow

@mgonto i really dont like Angular 2 so far. not going to judge it till i try writing and not going to bother till it locks down

LIKE

1



jbarket

@jbarket



Follow

I really, really want to be excited about Angular 2, but all the code samples look like decorators vomited all over them.

@UglyAsHell { }



131



94



```

/**
 * Simplest possible template in AngularJs-ISH style
 *
 * @param {String} template - template string
 * @param {Object} ctx - template context
 * @param {Object} eventHandlerObject - object that will be used as "this" in event handling
 * @returns {Node} returns dom node element
 */
export default function angularish(template, ctx, eventHandlerObject) {
  var node;
  var container = document.createElement('div');

  container.innerHTML = template;

  var walker = document.createTreeWalker(container, NodeFilter.SHOW_ELEMENT, null, false);
  while (node = walker.nextNode()) {

    // inheritance of context
    node.ctx = node.ctx || node.parentNode.ctx || ctx;

    // ng-scope allows you to change scope of the node (new scope can be any property of old
scope)
    if (node.getAttribute('ng-scope')) {

      node.ctx = _getValue(node.ctx, node.getAttribute('ng-scope'));

    }

    // ng-loop will repeat first child (TODO: repeat content) and assign correct context
    if (node.getAttribute('ng-loop')) {

```

```

/**
 * Simplest possible template in AngularJs-ISH style
 *
 * @param {String} template - template string
 * @param {Object} ctx - template context
 * @param {Object} eventHandlerObject - object that will be used as "this" in event handling
 * @returns {Node} returns dom node element
 */
export default function angularish(template, ctx, eventHandlerObject) {
  var node;
  var container = document.createElement('div');

  container.innerHTML = template;

  var walker = document.createTreeWalker(container, NodeFilter.SHOW_ELEMENT, null, false);
  while (node = walker.nextNode()) {

    // inheritance of context
    node.ctx = node.ctx || node.parentNode.ctx || ctx;

    // ng-scope allows you to change scope of the node (new scope can be any property of old
scope)
    if (node.getAttribute('ng-scope')) {

      node.ctx = _getValue(node.ctx, node.getAttribute('ng-scope'));

    }

    // ng-loop will repeat first child (TODO: repeat content) and assign correct context
    if (node.getAttribute('ng-loop')) {

```

```

/**
 * Simplest possible template in AngularJs-ISH style
 *
 * @param {String} template - template string
 * @param {Object} ctx - template context
 * @param {Object} eventHandlerObject - object that will be used as "this" in event handling
 * @returns {Node} returns dom node element
 */
export default function angularish(template, ctx, eventHandlerObject) {
  var node;
  var container = document.createElement('div');

  container.innerHTML = template;

  var walker = document.createTreeWalker(container, NodeFilter.SHOW_ELEMENT, null, false);
  while (node = walker.nextNode()) {

    // inheritance of context
    node.ctx = node.ctx || node.parentNode.ctx || ctx;

    // ng-scope allows you to change scope of the node (new scope can be any property of old
scope)
    if (node.getAttribute('ng-scope')) {

      node.ctx = _getValue(node.ctx, node.getAttribute('ng-scope'));

    }

    // ng-loop will repeat first child (TODO: repeat content) and assign correct context
    if (node.getAttribute('ng-loop')) {

```

```
node.value = _getValue(node.ctx, node.getAttribute('ng-value'));

}
// ng-selected will set selected attribute depending on true-finess of value
if (node.getAttribute('ng-selected')) {

    var selected = _getValue(node.ctx, node.getAttribute('ng-selected'));
    if (selected) {
        node.setAttribute('selected', 'yes');
    }

}

// ng-text will assign text to node no need for escaping
if (node.getAttribute('ng-text')) {

    node.innerText = _getValue(node.ctx, node.getAttribute('ng-text'));

}

// ng-class will simply assign class from defined property
if (node.getAttribute('ng-class')) {

    var classVal = _getValue(node.ctx, node.getAttribute('ng-class'));
    if (classVal) {
        node.className += ' ' + classVal;
    }

}

// ng-show shows elements depending on true-finess of the value
if (node.getAttribute('ng-show')) {
```

```
scope)
// ng-scope allows you to change scope of the node (new scope can be any property of old
scope)
if (node.getAttribute('ng-scope')) {
    node.ctx = _getValue(node.ctx, node.getAttribute('ng-scope'));
}
// ng-loop will repeat first child (TODO: repeat content) and assign correct context
if (node.getAttribute('ng-loop')) {
    var child = node.children[0];
    var array = _getValue(node.ctx, node.getAttribute('ng-loop')) || [];
    node.removeChild(child);
    array.forEach((item) => {
        child = child.cloneNode(true);
        child.ctx = item;
        node.appendChild(child);
    });
}
// ng-value will assign value to node
if (node.getAttribute('ng-value')) {
    node.value = _getValue(node.ctx, node.getAttribute('ng-value'));
}
// ng-selected will set selected attribute depending on true-finess of value
if (node.getAttribute('ng-selected')) {
```

```

// ng-change will add "change" event handler
if (node.getAttribute('ng-change')) {
  // closure to rescue
  ((node)=> {
    node.addEventListener('change', (event) => {
      eventHandlerObject[node.getAttribute('ng-change')]
        .bind(eventHandlerObject)(node.ctx, event);
    }, true);
  })(node);
}
// ng-click will add "click" event handler
if (node.getAttribute('ng-click')) {
  // closure to rescue
  ((node)=> {
    node.addEventListener('click', (event) => {
      eventHandlerObject[node.getAttribute('ng-click')]
        .bind(eventHandlerObject)(node.ctx, event);
    }, true);
  })(node);
}

return container;
}

function _getValue(ctx, attrVal) {
  if (attrVal === 'self') {
    return ctx;
  }
}

```



```

}
// ng-hide shows elements depending on false-iness of the value
if (node.getAttribute('ng-hide')) {

    var isHidden = __getValue(node.ctx, node.getAttribute('ng-hide'));
    if (isHidden) {
        node.style.display = 'none';
    }

}

// ng-change will add "change" event handler
if (node.getAttribute('ng-change')) {
    // closure to rescue
    ((node)=> {
        node.addEventListener('change', (event) => {
            eventHandlerObject[node.getAttribute('ng-change')]
                .bind(eventHandlerObject)(node.ctx, event);
        }, true);
    })(node);
}

// ng-click will add "click" event handler
if (node.getAttribute('ng-click')) {
    // closure to rescue
    ((node)=> {
        node.addEventListener('click', (event) => {
            eventHandlerObject[node.getAttribute('ng-click')]
                .bind(eventHandlerObject)(node.ctx, event);
        }, true);
    })(node);
}

```

[]

()

[()]

[property]='value' -> property='value'

()

[()]

[property]='value' -> property='value'

(event)='handler()' -> on-event='handler()'

[()]

[property]='value' -> property='value'

(event)='handler()' -> on-event='handler()'

[(target)]='value' -> on-change='update()'
-> target='value'

bind-property='value' -> property='value'

(event)='handler()' -> on-event='handler()'

[(target)]='value' -> on-change='update()'
-> target='value'

bind-property='value' -> property='value'

on-event='handler()' -> on-event='handler()'

[(target)]= 'value' -> on-change='update()'
-> target='value'

bind-property='value' -> property='value'

on-event='handler()' -> on-event='handler()'

bindon-prop='value' -> on-change='update()'
-> target='value'


```
<hero-detail *ngIf="currentHero"  
  [hero]="currentHero"/>
```

```
<hero-detail  template="ngIf:currentHero"  
  [hero]="currentHero"/>
```

```
<template [ngIf]="currentHero">  
  <hero-detail [hero]="currentHero"></hero-detail>  
</template>
```

System.js & JSPM & System.js Builder

<http://plnkr.co/>

System.js & JSPM & System.js Builder

```
<title>angular2 playground</title>
<link rel="stylesheet" href="style.css" />
<script src="https://code.angularjs.org/2.0.0-beta.17/
angular2-polyfills.js"></script>
<script src="https://code.angularjs.org/tools/system.js"></
script>
<script src="https://code.angularjs.org/tools/
typescript.js"></script>
<script src="config.js"></script>
<script>
    System.import('app')
        .catch(console.error.bind(console));
</script>
</head>
```

System.js & JSPM & System.js Builder

```
<title>angular2 playground</title>
<link rel="stylesheet" href="style.css" />
<script src="https://code.angularjs.org/2.0.0-beta.17/
angular2-polyfills.js"></script>
<script src="https://code.angularjs.org/tools/system.js"></
script>
<script src="https://code.angularjs.org/tools/
typescript.js"></script>
<script src="config.js"></script>
<script>
    System.import('app')
    .catch(console.error.bind(console)) ;
</script>
</head>
```

```

System.config({
  //use typescript for compilation
  transpiler: 'typescript',
  //typescript compiler options
  typescriptOptions: {
    emitDecoratorMetadata: true
  },
  //map tells the System loader where to look for things
  map: {
    app: './src',
    '@angular': 'https://npmcdn.com/@angular',
    'rxjs': 'https://npmcdn.com/rxjs@5.0.0-beta.6'
  },
  //packages defines our app package
  packages: {
    app: {
      main: './main.ts',
      defaultExtension: 'ts'
    },
    '@angular/core': {
      main: 'core.umd.js',
      defaultExtension: 'js'
    },
    '@angular/compiler': {
      main: 'compiler.umd.js',
      defaultExtension: 'js'
    },
    '@angular/common': {
      main: 'common.umd.js',
      defaultExtension: 'js'
    },
    '@angular/platform-browser-dynamic': {
      main: 'platform-browser-dynamic.umd.js',
      defaultExtension: 'js'
    },
    '@angular/platform-browser': {
      main: 'platform-browser.umd.js',
      defaultExtension: 'js'
    },
    rxjs: {
      defaultExtension: 'js'
    }
  }
});

```

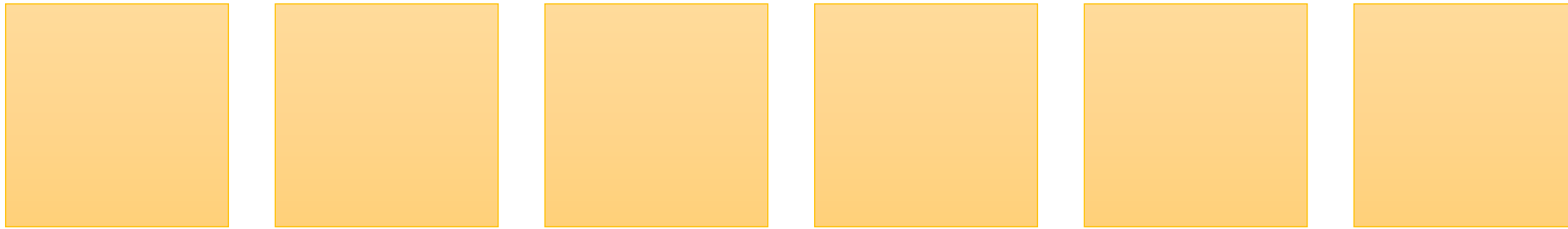
44 строки конфига

```
System.config({
  //use typescript for compilation
  transpiler: 'typescript',
  //typescript compiler options
  typescriptOptions: {
    emitDecoratorMetadata: true
  },
  //map tells the System loader where to look for things
  map: {
    app: './src',
    '@angular': 'https://npmcdn.com/@angular',
    'rxjs': 'https://npmcdn.com/rxjs@5.0.0-beta.6'
  },
  //packages defines our app package
  packages: {
    app: {
      main: './main.ts',
      defaultExtension: 'ts'
    },
    '@angular/core': {
      main: 'core.umd.js',
      defaultExtension: 'js'
    },
    '@angular/compiler': {
      main: 'compiler.umd.js',
      defaultExtension: 'js'
    },
    '@angular/common': {
      main: 'common.umd.js',
      defaultExtension: 'js'
    },
    '@angular/platform-browser-dynamic': {
      main: 'platform-browser-dynamic.umd.js',
      defaultExtension: 'js'
    },
    '@angular/platform-browser': {
      main: 'platform-browser.umd.js',
      defaultExtension: 'js'
    },
    rxjs: {
      defaultExtension: 'js'
    }
  }
});
```


Не хуже чем Webpack

- Официальный Angular QuickStart репозиторий
- Angular CLI
- Yoman / Slush - генераторы

TypeScript



TypeScript



TypeScript



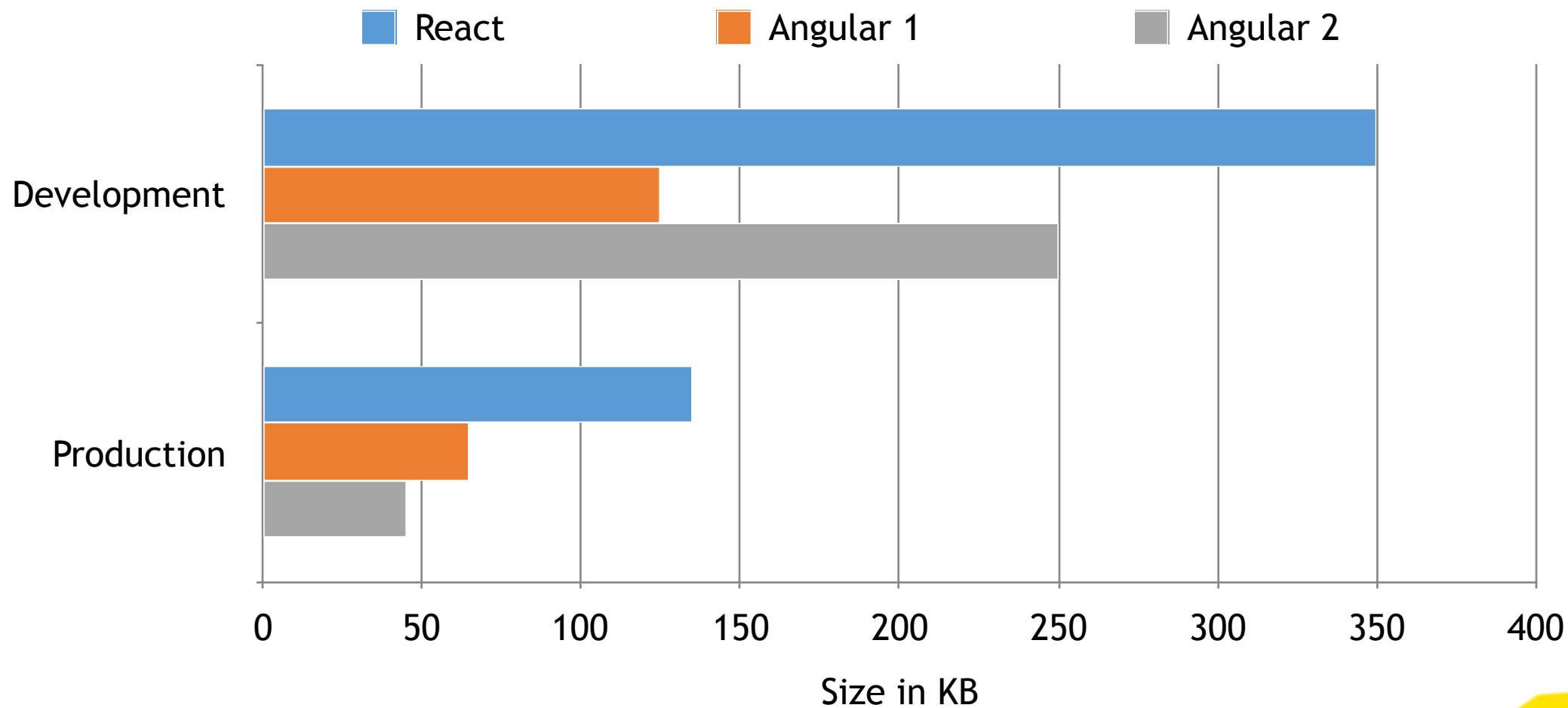
TypeScript

- .Net, Java, Scala background
- SOLID, Design Patterns
- 1.8.10
- Poor documentation - «google search ftw»

Глава №3 - Добыча



Размер



Размер - Angular 2 Router

Lazy
Loading




```
@Routes([
  {path: 'user/:id', component: "UserCmp"},
  {path: 'role/:id', component: "RoleCmp"},
])
class MyApp{}
```

```
src
  my-app.ts
  +user-cmp
    user-cmp.ts
  +role-cmp
    role-cmp.ts
```

Размер - Angular 2 Router

This chapter describes the *deprecated beta* Component Router which is replaced by the *release candidate* Component Router. We are documenting that now.

CSS

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

@Component({
  selector: 'my-app',
  providers: [],
  styles: [`
    h2 { color: red; }
  `],
  template: `
    <div>
      <h2>Hello {{name}}</h2>
    </div>
  `,
  directives: []
})
export class App {
  constructor() {
    this.name = 'Angular2 (Release Candidate!)'
  }
}
```

Hello Angular2 (Release Candidate!)

CSS

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

@Component({
  selector: 'my-app',
  providers: [],
  styles: [`
    body { color: red; }
  `],
  template: `
    <div>
      <h2>Hello {{name}}</h2>
    </div>
  `,
  directives: []
})
export class App {
  constructor() {
    this.name = 'Angular2 (Release Candidate!)'
  }
}
```

Hello Angular2 (Release Candidate!)

CSS

```
<script>
  System.import('app')
    .catch(console.error.bind(console));
</script>
<style>body[_ngcontent-sfq-1] { color: red; }</style> == $0
</head>
▼<body>
  ▼<my-app _ngghost-sfq-1>
    ▼<div _ngcontent-sfq-1>
      <h2 _ngcontent-sfq-1>Hello Angular2 (Release Candidate!)</h2>
    </div>
```

Hello Angular2 (Release Candidate!)

Speed - templates

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

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  template: `
    <h1>
      {{title}}
    </h1>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  title = 'project-name works!';
}
```

```

new jit_StaticNodeDebugInfo0([],null,{}),
new jit_StaticNodeDebugInfo0([],null,{}))
]
;
var renderType_PROJECTNAMEAppComponent = null;
function _View_PROJECTNAMEAppComponent0(viewUtils,parentInjector,declarationEl) {
  var self = this;
  jit_DebugAppView1.call(this,
_View_PROJECTNAMEAppComponent0,renderType_PROJECTNAMEAppComponent,jit_ViewType_COMPONENT2,viewUtils,parentInjector,declarationEl,jit_ChangeDetectionStrategy_CheckAlways3,nodeDebugInfos_PROJECTNAMEAppComponent0);
}
_View_PROJECTNAMEAppComponent0.prototype = Object.create(jit_DebugAppView1.prototype);
_View_PROJECTNAMEAppComponent0.prototype.createInternal = function(rootSelector) {
  var self = this;
  var parentRenderNode = self.renderer.createViewRoot(self.declarationAppElement.nativeElement);
  self._el_0 = self.renderer.createElement(parentRenderNode,'h1',self.debug(0,0,0));
  self._text_1 = self.renderer.createText(self._el_0,"",self.debug(1,0,4));
  self._text_2 = self.renderer.createText(parentRenderNode,'\n',self.debug(2,2,5));
  self._expr_0 = jit_uninitialized4;
  self.init([], [
    self._el_0,
    self._text_1,
    self._text_2
  ], [], []);
  return null;
};
_View_PROJECTNAMEAppComponent0.prototype.detectChangesInternal = function(throwOnChange) {
  var self = this;
  self.detectContentChildrenChanges(throwOnChange);
  self.debug(1,0,4);
  var currVal_0 = jit_interpolate5(1, '\n', self.context.title, '\n');

```



```

new jit_StaticNodeDebugInfo0([],null,{}),
new jit_StaticNodeDebugInfo0([],null,{}),
new jit_StaticNodeDebugInfo0([],null,{}),
]
;
var renderType_PROJECTNAMEAppComponent = null;
function _View_PROJECTNAMEAppComponent0(viewUtils,parentInjector,declarationEl) {
  var self = this;
  jit_DebugAppView1.call(this,
_View_PROJECTNAMEAppComponent0,renderType_PROJECTNAMEAppComponent,jit_ViewType_COMPONENT2,viewUtils,parentInjector,declarationEl,jit_ChangeDetectionStrategy_CheckAlways3,nodeDebugInfos_PROJECTNAMEAppComponent0);
}
_View_PROJECTNAMEAppComponent0.prototype = Object.create(jit_DebugAppView1.prototype);
_View_PROJECTNAMEAppComponent0.prototype.createInternal = function(rootSelector) {
  var self = this;
  var parentRenderNode = self.renderer.createViewRoot(self.declarationAppElement.nativeElement);
  self._el_0 = self.renderer.createElement(parentRenderNode,'h1',self.debug(0,0,0));
  self._text_1 = self.renderer.createText(self._el_0,"",self.debug(1,0,4));
  self._text_2 = self.renderer.createText(parentRenderNode,'\n',self.debug(2,2,5));
  self._expr_0 = jit_uninitialized4;
  self.init([], [
    self._el_0,
    self._text_1,
    self._text_2
  ], [], []);
  return null;
};
_View_PROJECTNAMEAppComponent0.prototype.detectChangesInternal = function(throwOnChange) {
  var self = this;
  self.detectContentChildrenChanges(throwOnChange);
  self.debug(1,0,4);
  var currVal_0 = jit_interpolate5(1 '\n ' self.context.title '\n');

```

self.renderer.createElement

self.renderer.createText

Как построить DOM

Роман Дворнов

Speed immutability

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

@Pipe({ name: 'isOdd' })
export class IsOddPipe implements PipeTransform {
  transform(array:any[]) { return array.filter(item => item.isOdd); }
}

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  pipes: [IsOddPipe],
  template: `
    <button (click)="add()">add</button>
    <div>
      <div *ngFor="let item of list | isOdd">
        {{ item.name }}
      </div>
    </div>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  list = []
  add() {
    this.list.push({ name: 'test', isOdd: !(this.list.length % 2) })
  }
}
```

Speed immutability

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

@Pipe({ name: 'isOdd' })
export class IsOddPipe implements PipeTransform {
  transform(array:any[]) { return array.filter(item => item.isOdd); }
}

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  pipes: [IsOddPipe],
  template: `
    <button (click)="add()">add</button>
    <div>
      <div *ngFor="let item of list | isOdd">
        {{ item.name }}
      </div>
    </div>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  list = []
  add() {
    this.list.push({ name: 'test', isOdd: !(this.list.length % 2) })
  }
}
```

Speed immutability

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

@Pipe({ name: 'isOdd' })
export class IsOddPipe implements PipeTransform {
  transform(array:any[]) { return array.filter(item => item.isOdd); }
}

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  pipes: [IsOddPipe],
  template: `
    <button (click)="add()">add</button>
    <div>
      <div *ngFor="let item of list | isOdd">
        {{ item.name }}
      </div>
    </div>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  list = []
  add() {
    this.list.push({ name: 'test', isOdd: !(this.list.length % 2) })
  }
}
```

Speed immutability

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

@Pipe({ name: 'isOdd', is_pure: false })
export class IsOddPipe implements PipeTransform {
  transform(array: any[]) { return array.filter(item => item.isOdd); }
}

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  pipes: [IsOddPipe],
  template: `
    <button (click)="add()">add</button>
    <div>
      <div *ngFor="let item of list | isOdd">
        {{ item.name }}
      </div>
    </div>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  list = []
  add() {
    this.list.push({ name: 'test', isOdd: !(this.list.length % 2) })
  }
}
```

Speed immutability

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

@Pipe({ name: 'isOdd' })
export class IsOddPipe implements PipeTransform {
  transform(array:any[]) { return array.filter(item => item.isOdd); }
}

@Component({
  moduleId: module.id,
  selector: 'project-name-app',
  pipes: [IsOddPipe],
  template: `
    <button (click)="add()">add</button>
    <div>
      <div *ngFor="let item of list | isOdd">
        {{ item.name }}
      </div>
    </div>
  `,
  styleUrls: ['project-name.component.css']
})
export class PROJECTNAMEAppComponent {
  list = []
  add() {
    this.list = this.list.splice().filter((i) => i % 2)
  }
}
```


Speed - zone.js

```
Zone.fork().run(function () {  
  zone.inTheZone = true;  
  
  setTimeout(function () {  
    console.log('in the zone: ' + !!zone.inTheZone);  
  }, 0);  
});
```

```
console.log('in the zone: ' + !!zone.inTheZone);
```

```
'in the zone: false'  
'in the zone: true'
```

Speed - zone.js

```
Zone.fork().run(function () {  
  zone.inTheZone = true;  
  
  setTimeout(function () {  
    console.log('in the zone: ' + !!zone.inTheZone);  
  }, 0);  
});
```

```
console.log('in the zone: ' + !!zone.inTheZone);
```

```
'in the zone: false'  
'in the zone: true'
```

TypeScript OOP

```
class GenericService<T> {  
  items: Array<T> = []  
  
  addItem(item: T) {  
    this.items.push(item)  
  }  
}
```

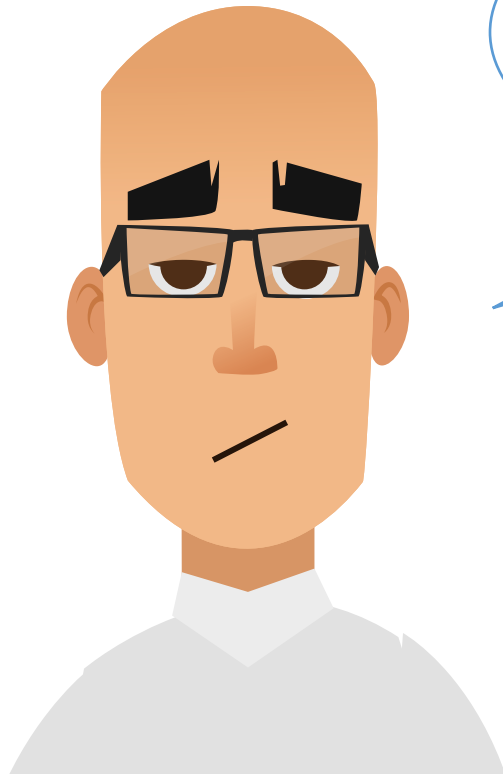
```
interface User {  
  id: number,  
  name: string  
}
```

```
interface Creatives {  
  type: string,  
  value: string  
}
```

TypeScript OOP

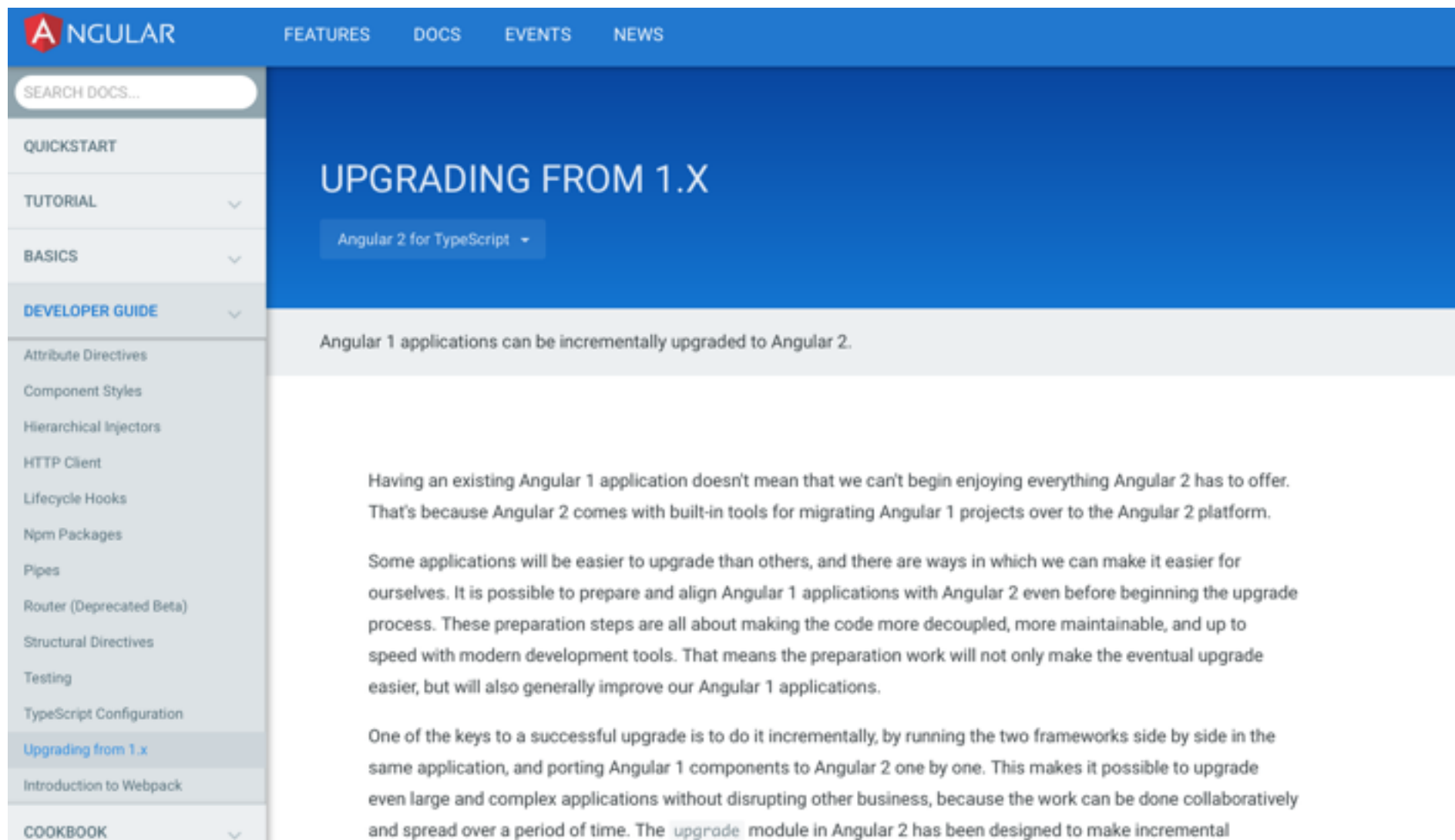
```
var s = new GenericService<User>();  
s.addItem({  
    id: 1, name: 'asda'  
});  
  
s.addItem({  
    type: 'asda'    // will fail  
})
```

Глава №4 - Первые потери



... а вот этого я не ожидал

Потеря почти всей кодовой базы



The screenshot shows the Angular website's navigation and main content area. The top navigation bar includes the Angular logo and links for FEATURES, DOCS, EVENTS, and NEWS. A search bar is located below the navigation bar. The left sidebar contains a list of documentation topics, with 'UPGRADING FROM 1.X' highlighted. The main content area features a large blue header with the title 'UPGRADING FROM 1.X' and a sub-header 'Angular 2 for TypeScript'. Below this, a paragraph states: 'Angular 1 applications can be incrementally upgraded to Angular 2.' The main text area contains three paragraphs discussing the upgrade process, including the use of the 'upgrade' module in Angular 2.

ANGULAR

FEATURES DOCS EVENTS NEWS

SEARCH DOCS...

QUICKSTART

TUTORIAL

BASICS

DEVELOPER GUIDE

Attribute Directives

Component Styles

Hierarchical Injectors

HTTP Client

Lifecycle Hooks

Npm Packages

Pipes

Router (Deprecated Beta)

Structural Directives

Testing

TypeScript Configuration

Upgrading from 1.x

Introduction to Webpack

COOKBOOK

UPGRADING FROM 1.X

Angular 2 for TypeScript

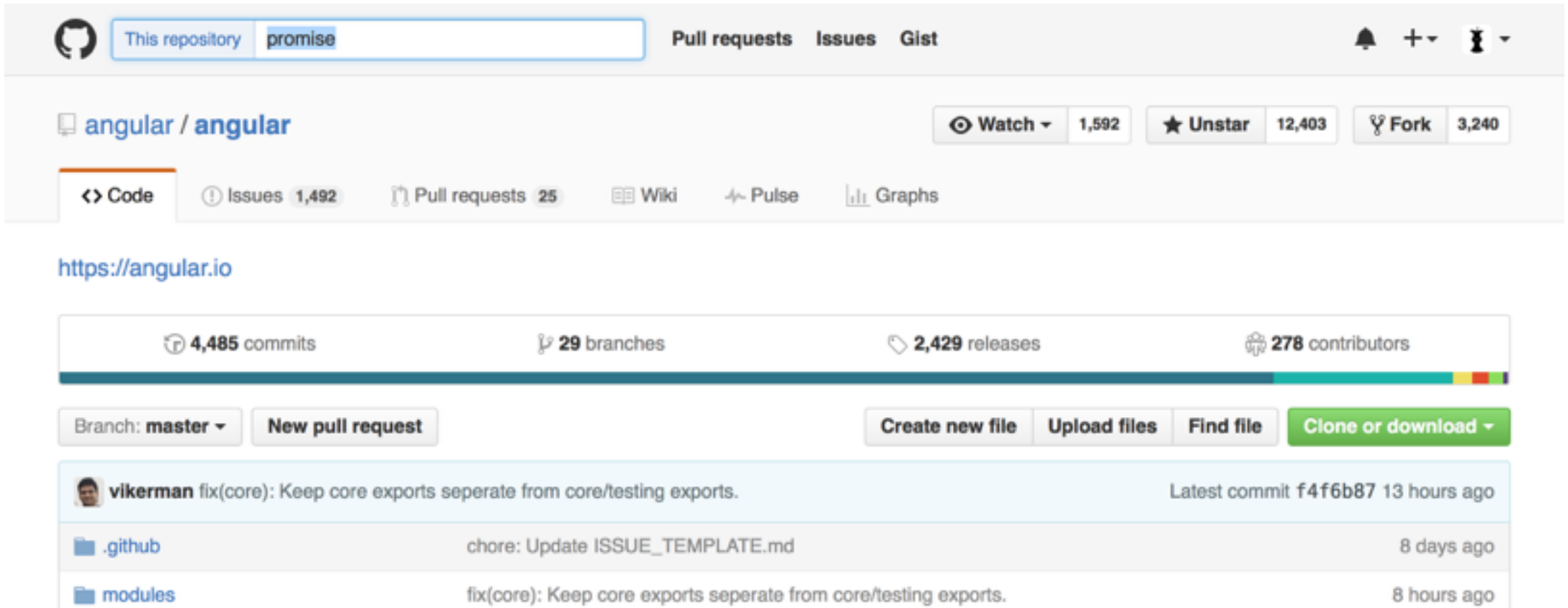
Angular 1 applications can be incrementally upgraded to Angular 2.

Having an existing Angular 1 application doesn't mean that we can't begin enjoying everything Angular 2 has to offer. That's because Angular 2 comes with built-in tools for migrating Angular 1 projects over to the Angular 2 platform.

Some applications will be easier to upgrade than others, and there are ways in which we can make it easier for ourselves. It is possible to prepare and align Angular 1 applications with Angular 2 even before beginning the upgrade process. These preparation steps are all about making the code more decoupled, more maintainable, and up to speed with modern development tools. That means the preparation work will not only make the eventual upgrade easier, but will also generally improve our Angular 1 applications.

One of the keys to a successful upgrade is to do it incrementally, by running the two frameworks side by side in the same application, and porting Angular 1 components to Angular 2 one by one. This makes it possible to upgrade even large and complex applications without disrupting other business, because the work can be done collaboratively and spread over a period of time. The `upgrade` module in Angular 2 has been designed to make incremental

Promise -> RXJS



The screenshot shows the GitHub interface for the 'angular/angular' repository. At the top, there's a search bar with 'promise' entered. Below the repository name, there are buttons for 'Watch' (1,592), 'Unstar' (12,403), and 'Fork' (3,240). The 'Code' tab is selected, showing a list of recent commits. The first commit is by 'vikerman' with the message 'fix(core): Keep core exports separate from core/testing exports.' and a commit hash of 'f4f6b87' from 13 hours ago. Below this, two other commits are listed: 'chore: Update ISSUE_TEMPLATE.md' from 8 days ago and another 'fix(core): Keep core exports separate from core/testing exports.' from 8 hours ago.

angular / angular

Watch 1,592 Unstar 12,403 Fork 3,240

Code Issues 1,492 Pull requests 25 Wiki Pulse Graphs

<https://angular.io>

4,485 commits 29 branches 2,429 releases 278 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

vikerman fix(core): Keep core exports separate from core/testing exports. Latest commit f4f6b87 13 hours ago

.github chore: Update ISSUE_TEMPLATE.md 8 days ago

modules fix(core): Keep core exports separate from core/testing exports. 8 hours ago

Promise -> RXJS

The screenshot shows the GitHub repository page for 'angular/angular'. At the top, the repository name is displayed with icons for 'Watch' (1,592), 'Unstar' (12,403), and 'Fork' (3,240). Below this is a navigation bar with links for 'Code', 'Issues' (1,492), 'Pull requests' (25), 'Wiki', 'Pulse', and 'Graphs'. On the left side, there is a sidebar with 'Code' and 'Issues' links. The main content area features a search bar with the text 'promise' and a 'Search' button. Below the search bar, a message states: 'We couldn't find any code matching 'promise''. Below this message, it says 'You could try an [advanced search](#).'

Promise -> RXJS

```
import {Http, HTTP_PROVIDERS} from 'angular2/http';
@Component({
  selector: 'http-app',
  viewProviders: [HTTP_PROVIDERS],
  templateUrl: 'people.html'
})
class PeopleComponent {
  constructor(http: Http) {
    http.get('people.json')
      .map(res => res.json())
      .subscribe(people => this.people = people);
  }
}
```

Promise -> RXJS

```
import {Http, HTTP_PROVIDERS} from 'angular2/http';
@Component({
  selector: 'http-app',
  viewProviders: [HTTP_PROVIDERS],
  templateUrl: 'people.html'
})
class PeopleComponent {
  constructor(http: Http) {
    http.get('people.json')
      .map(res => res.json())
      .subscribe(people => this.people = people);
  }
}
```

Promise -> RXJS

```
import {Http, HTTP_PROVIDERS} from 'angular2/http';
@Component({
  selector: 'http-app',
  viewProviders: [HTTP_PROVIDERS],
  templateUrl: 'people.html'
})
class PeopleComponent {
  constructor(http: Http) {
    http.get('people.json')
      .map(res => res.json())
      .subscribe(people => this.people = people);
  }
}
```

RXJS

Expert to Expert: Brian Beckman and Erik Meijer - Inside the .NET Reactive Framework (Rx)

Опубликовано: июл 09, 2009 в 11:36

Автор: **Charles**

★★★★★ (24) | просмотров: 135,097 | комментариев: 57

Средний: 5



RXJS

```
interface IObservable<T>
{
    IDisposable Subscribe(IObserver observer);
}
```

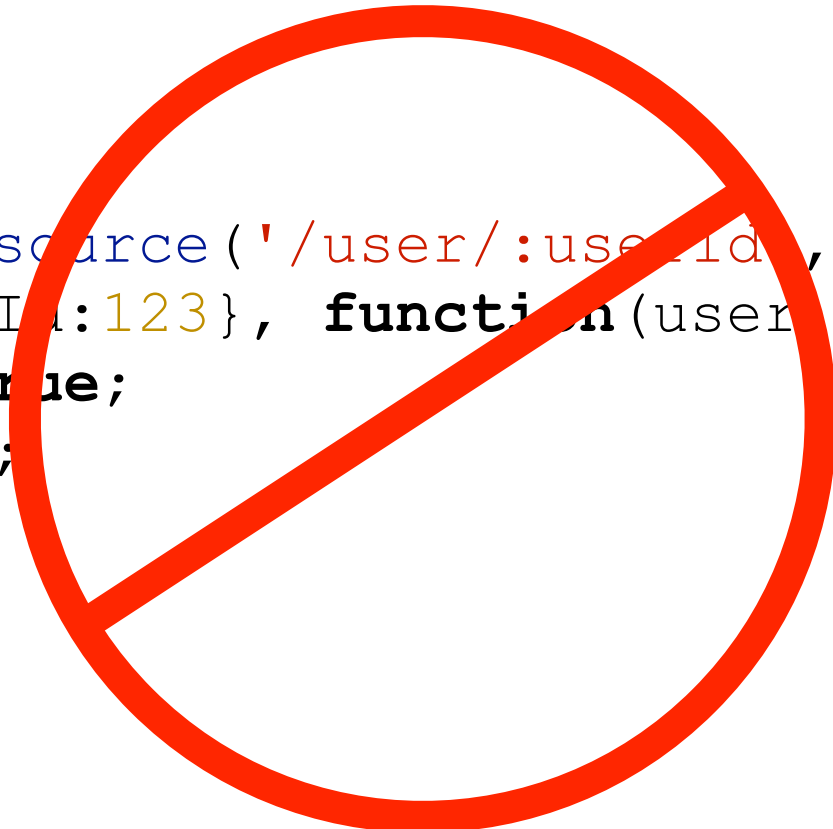
```
interface IObserver<T>
{
    void OnCompleted();
    void OnNext(T value);
    void OnError(Exception e);
}
```

ngResources

```
var User = $resource('/user/:userId', {userId: '@id'});  
User.get({userId: 123}, function(user) {  
    user.abc = true;  
    user.$save();  
});
```

ngResources

```
var User = $resource('/user/:userId', {userId: '@id'});  
User.get({userId: 123}, function(user) {  
    user.abc = true;  
    user.$save();  
});
```



Встроенные паттерны канули в небытие!

- 1) component
- 2) directive
- 3) filter
- 4) service
- 5) provider
- 6) constant
- 7) config
- 8) run
- 9) module

Встроенные паттерны канули в небытие!

- 1) component
- 2) template
- 3) directive
- 4) route
- 5) pipe
- 6) service *

Формы

1) [(ngModel)]

Формы

1) [(ngModel)]

2) ng-valid | ng-invalid | ng-dirty | ng-pristine | ng-touched | ng-untouched

Формы

- 1) [(ngModel)]
- 2) ng-valid | ng-invalid | ng-dirty | ng-pristine | ng-touched | ng-untouched
- 3) FormModel + FormBuilder

Формы

- 1) [(ngModel)]
- 2) ng-valid | ng-invalid | ng-dirty | ng-pristine | ng-touched | ng-untouched
- 3) FormModel + FormBuilder
- 4) Валидация не стала легче

Глава №5 - Happy End



Мы уже переехали на Angular 2?

HET

Почему?

1) Потому что Angular 1 не так уж и плох, а если задуматься ...

Почему?

- 1) Потому что Angular 1 не так уж и плох, а если задуматься ...
- 2) Потому что React 15 не так уж и плох, а если задуматься ...

Почему?

- 1) Потому что Angular 1 не так уж и плох, а если задуматься ...
- 2) Потому что React 15 не так уж и плох, а если задуматься ...
- 3) Потому что Ember не так уж и плох, а если задуматься ...

Почему?

- 1) Кодовая база
- 2) Уровень вхождения
- 3) Незаконченность*

Почему?

- 1) Кодовая база
- 2) Уровень вхождения
- 3) Незаконченность*

Есть ли надежда?

Наши шаги

- 1) TypeScript OOP - e2e tests
- 2) Angular 1.4.x -> 1.5.x
- 3) AutoNgConverter

О чем мы не поговорили?

Progressive Web Apps

Native

- Ionic Framework,
- NativeScript
- React Native.

Desktop

- Electron

Universal

- node.js,
- .NET,
- PHP

Dependency Injection

Angular CLI

IDEs

Testing

- patched Karma, Protractor

Animation

Accessibility

Developer Tools

- Redux (ngrx / ng2-redux)
- FLUX
- MV* (MVC, MVP, MVVM)
- MALEVICH (COD.js)

Приятного аппетита!

Tweeter: #Ai_boy

Gitter: aiboy



**Frontend
Conf**



<http://bit.ly/1XP0dEh>

