Breizh C@mp
Mix de technologies

# Angular 2
# Le reveil de la force

**#BzhCmp #Ng2BzhCmp**

#Ng2BzhCmp #BzhCmp

## Nicolas Pennec

@NicoPennec

Full-Stack Web Developer

Co-Fondateur de @RennesJS

#Ng2BzhCmp #BzhCmp

# Grégory Houllier

@ghoullier

Architecte Web, passionné par le Front-End

Co-Fondateur de @RennesJS

#Ng2BzhCmp #BzhCmp

@RennesJS

rennesjs.org

meetup.com/RennesJS/

RENNES JS USER GROUP

Meetup le dernier jeudi de chaque mois

25/06/2015 à Epitech

## Angular 1.X

- Framework JS MV*
- Projet open-source porté par Google
- Version 1.X très populaire, 1er sur GitHub (39K stars)
- Concepts :
  - Orienté Single Page Application (SPA)
  - Étendre le langage HTML (directives)
  - Data Binding bi-directionnel
  - Dependency Injection, $scope, ...

# Annonce d'Angular 2

**Octobre 2014:**

- Première annonce
- Rupture de concepts
- Pas de rétrocompatibilité
- Nouveau langage (AtScript)

#Ng2BzhCmp #BzhCmp

# Annonce d'Angular 2

**Mars 2015**

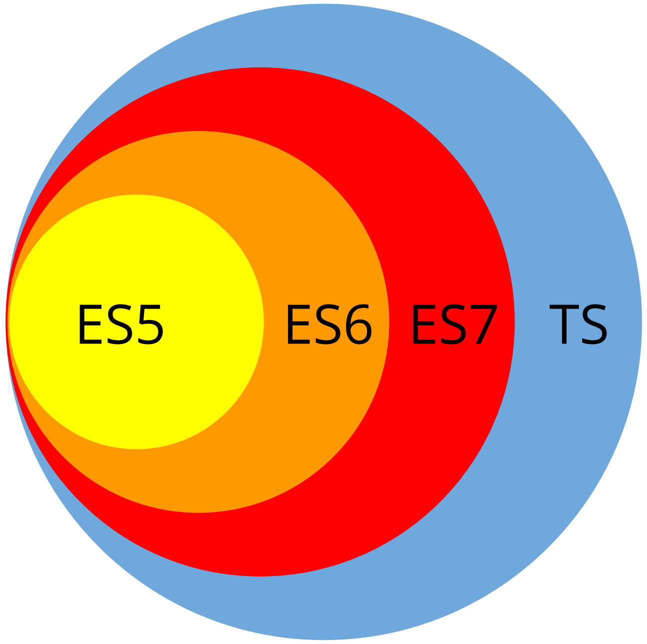
- Rétropédalage de Google
- Opération séduction des devs
- Roadmap de la branche 1.X
- Nouveau langage (TypeScript)

#Ng2BzhCmp #BzhCmp

## Philosophie

- Apprendre des erreurs d'Angular 1
  - 2-way data binding, dirty-checking, $scope, ...
- Se baser sur les futurs standards du web
  - WebComponents
  - ES6 / ES7 / TypeScript*
  - EverGreen Browsers
- Emulation due à la concurrence React et Ember
  - Amélioration des performances

# ES6, ES7, TypeScript



- ES5 : es5.github.io
- ES6 : git.io/es6features
- ES7 : draft
- TS : typescriptlang.org

- Traceur: github.com/google/traceur-compiler

#Ng2BzhCmp #BzhCmp

## TypeScript

- Surcouche à ES6/ES7
- Créé par Microsoft
- Collaboration avec Google
  - Merge avec AtScript
- Actuellement en version 1.5-beta

#Ng2BzhCmp #BzhCmp

# Exemple TypeScript

TS

```
import { Annotation } from 'angular2/angular2';
import { Api as ApiSpeakers } from './api/speakers';

@Annotation({
  property: 'value'
})
class Talk {

  speakers: Array<String>;
  thread: String;

  constructor(thread: String, api: ApiSpeakers) {
    this.thread = thread;
    api.get().then((speakers) => {
      this.speakers = speakers;
    });
  }
}
```

## Production Ready ?

**NO!**

# Production Ready ?

- Version Alpha "Developer Preview"
- Solution instable
- API changeante
- Documentation en cours
- Pas de Roadmap

## PoC Ready ?

**YES!**

#Ng2BzhCmp #BzhCmp

# Ce qui va changer ?

**Angular 1**

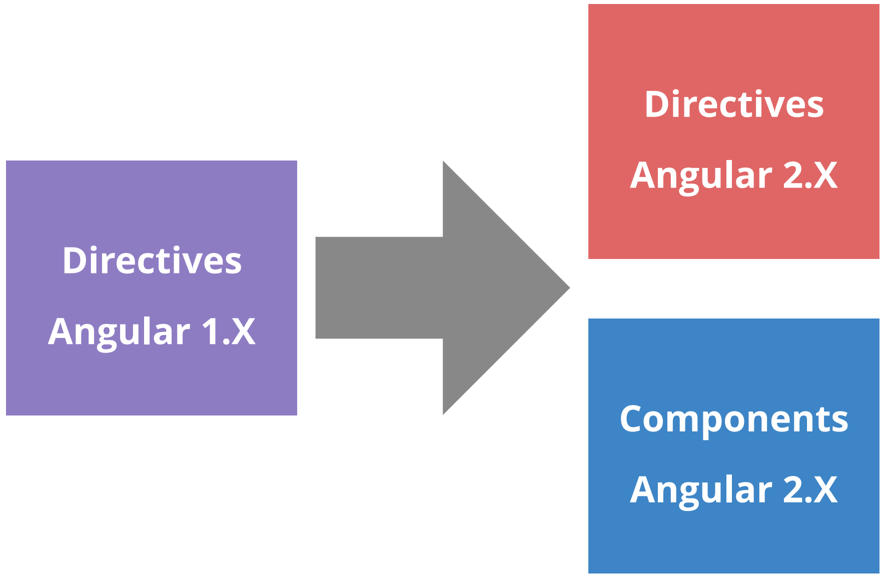
- Controller
- Service
- Module
- $scope
- jQlite
- Directive
- Dependency Injection

**Angular 2**

- ~~Controller~~
- ~~Service~~
- ~~Module~~
- ~~$scope~~
- ~~jQlite~~
- Directive
- Dependency Injection (TypeScript)
- Component
- Classes (ES6)

#Ng2BzhCmp #BzhCmp

# Directives/Components

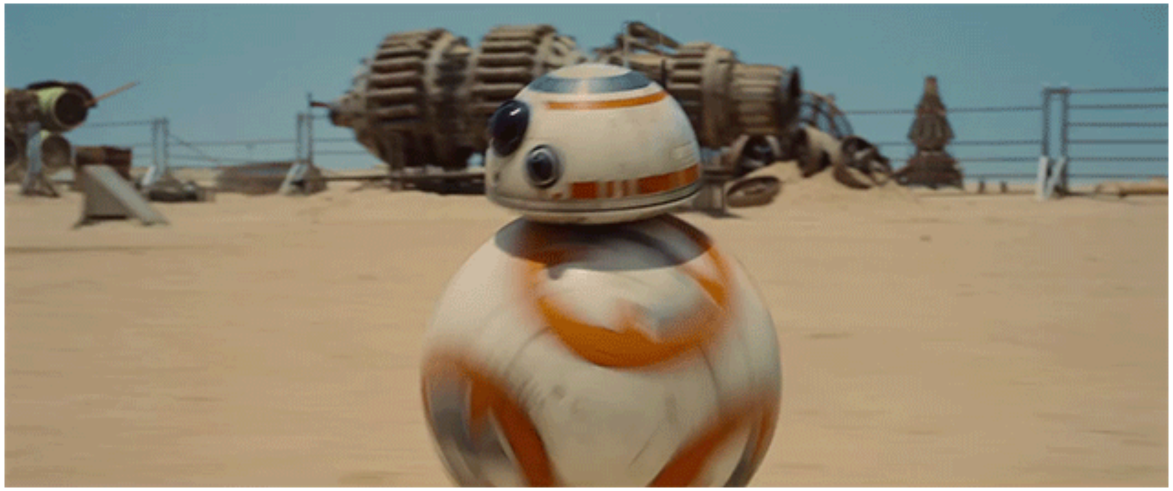


Etendre le DOM

Encapsule un template

#Ng2BzhCmp #BzhCmp

## A quoi ça va ressembler?



#Ng2BzhCmp #BzhCmp

# Bootstrap your code

HTML

```html
<!DOCTYPE html>
<html>
 <head>
   <title>Angular 2 - BreizhCamp</title>
   <script src="//github.jspm.io/jmcriffey/bower-traceur-runtime@0.0.90/traceur-runtime.js"></
   <script src="//jspm.io/system@0.16.js"></script>
   <script src="//code.angularjs.org/2.0.0-alpha.23/angular2.dev.js"></script>
 </head>
 <body>
   <my-component></my-component>
   <script type="module">
     import { bootstrap } from 'angular2/angular2';
     import { MyComponent } from 'app/my-component';

     bootstrap(MyComponent);
   </script>
 </body>
</html>
```

#Ng2BzhCmp #BzhCmp

# Component (1/3)

HTML

```html
<my-component></my-component>
```

TS

```ts
import {
  Component, View
} from 'angular2/angular2';

@Component({
  selector: 'my-component'
})
@View({
  template:
    '<div>Hello, BreizhCamp</div>'
})
export class MyComponent {

}
```

# Component (2/3)

HTML

```html
<my-component></my-component>
```

TS

```typescript
import {
  Component, View
} from 'angular2/angular2';

@Component({
  selector: 'my-component'
})
@View({
  template:
    '<div>Hello, {{message}}</div>'
})
export class MyComponent {
  constructor() {
    this.message = 'BreizhCamp';
  }
}
```

#Ng2BzhCmp #BzhCmp

# Component (3/3)

HTML

```html
<my-component msg="BreizhCamp">
</my-component>
```

TS

```typescript
import {
  Component, View
} from 'angular2/angular2';

@Component({
  selector: 'my-component',
  properties: {
    message: 'msg'
  }
})
@View({
  template:
    '<div>Hello, {{message}}</div>'
})
export class MyComponent {
}
```

# Directive (1/2)

HTML

```html
<div tooltip="Hello BreizhCamp">
  Hover Me!
</div>
```

TS

```typescript
import {
  Directive
} from 'angular2/angular2';

@Directive({
  selector: '[tooltip]',
  properties: {
    text: 'tooltip'
  },
  hostListeners: {
    mouseover: 'display()'
  }
})
export class Tooltip {
  display() {
    console.log(this.text);
  }
}
```

# Directive (2/2)

HTML

```html
<my-component msg="BreizhCamp">
</my-component>
```

TS

```typescript
import {
  Component, View
} from 'angular2/angular2';
import { Tooltip } from './tooltip';

@Component({
  selector: 'my-component',
  properties: {
    message: 'msg'
  }
})
@View({
  template:
    '<div tooltip="Yo!">Hi, {{message}}</div>',
  directives: [Tooltip]
})
export class MyComponent {}
```

## Templating (1/2)

### Interpolation

```
<div>Hello, {{username}}</div>
```

### Property binding / Event binding

```
<button [model]="message" (click)="hello(message)">
  Click Me!!
</button>
```

### Local variable (référence)

```
<audio-player #player></audio-player>
<button (click)="player.pause()">Pause</button>
```

# Templating (2/2)

## Whole template

```
<div *if="user">Hello, {{user.name}}</div>

// Équivalent à

<template if="user">
  <div>Hello, {{user.name}}</div>
</template>
```

```
<ul *if="list.length">
  <li *for="#item of list">
    {{item.title}}
  </li>
</ul>
```

#Ng2BzhCmp #BzhCmp

## Dependency Injection

```
import { MyService } from './my-service';

@Component({
  selector: 'my-component',
  injectables: [MyService]
})
@View({
  templateUrl: '/path/to/my-component.html'
})
class MyComponent {
  myService: MyService;

  constructor(myService: MyService) {
    this.myService = myService;
  }
  fetchData() {
    this.myService.get().then((list) => {
      console.log(list);
    });
  }
}
```

#Ng2BzhCmp #BzhCmp

## Migration 1.X vers 2.X ?

### Pas de rétrocompatibilité

### Mais comment anticiper ces changements?

- Classes ES6/TS pour la définition des services
- Modules ES6 pour structurer son code
- Utiliser le ngNewRouter qui ~~est~~ sera disponible en ~~1.4~~ 1.5
- Préférer la syntaxe *"controllerAs"* et *"bindToController"* pour les directives

#Ng2BzhCmp #BzhCmp

## Documentation

angular.io

#Ng2BzhCmp #BzhCmp

## Ressources

- github.com/angular/angular
- youtube.com/user/ngconfvideos
- angular-tips.com
- tryangular2.com
- egghead.io/technologies/angular2
- victorsavkin.com
- github.com/timjacobi/angular2-education

#Ng2BzhCmp #BzhCmp

## Conclusion

Angular 2 c'est comme le prochain Star Wars

tout le monde l'attend, mais personne ne sait si ça sera à la hauteur

#Ng2BzhCmp #BzhCmp