



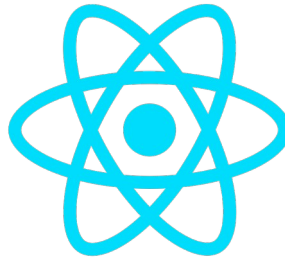
**Denis Apparicio**

# Présentation

- Permet de faire des applications cross-mobile
- Write once run every mobile

# Technologies Ionic

- Ionic est basé sur :
  - Angular/ React/Vue



- Apache cordova / Capacitor



# Introduction à Angular

# Historique

- Angular succède à AngularJS.
- Développé par la même équipe Google qu'AngularJS
- Refonte complète. Non compatible AngularJS
- Une attente longue de la version finale
  - Avril 2015 : version Alpha
  - Septembre 2016 : final release
- Activité :
  - Une version majeure tous les 6 mois
  - LTS 18 mois
- Angular 11 version active
- Plus d'informations : <https://angular.io/guide/releases>

# Compatibilité avec les navigateurs



**Mise à jour « automatique »**



**9**



**4.1**

**iOS**


**7.1**

# Introduction

- Framework basé sur les Web Component
- Pattern MVVM (Model-View View-Model)
- Technologies mis en œuvre
  - HTML5
  - TypeScript

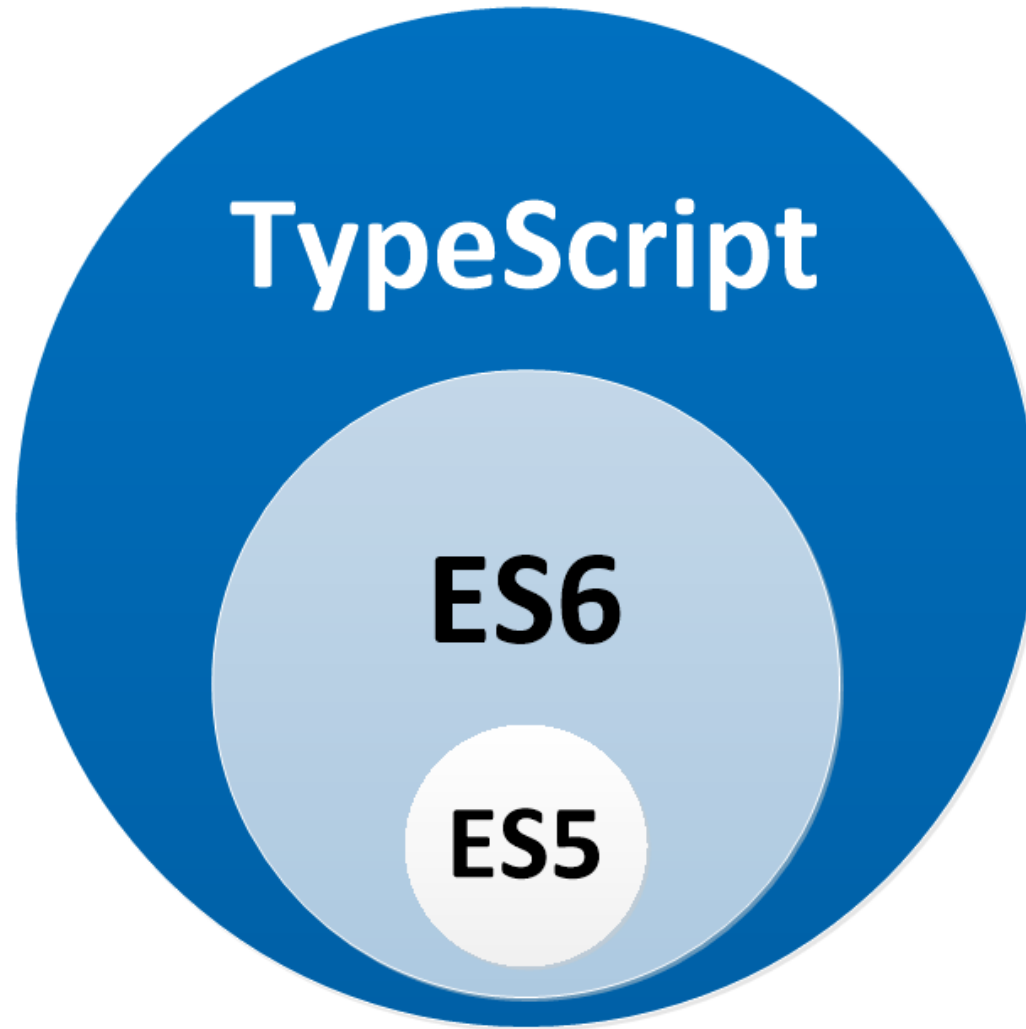
# TypeScript

## Introduction

- Historique
  - Octobre 2012 : Preview
  - Avril 2014 : release 1.0
  - Avril 2016 : release 2.2
  - Février 2018 : release 2.7.2
  - Mars 2021 : release 4.2.3
- Anders Hejlsberg fait partie des créateurs.
  - Compilateur Pascal, Bordland, C#
- TypeScript est un langage transcompilé
  - TypeScript  EcmaScript5, EcmaScript6



# TypeScript **Ts**



# TypeScript

## Installation

- Installation de npm  $\Rightarrow$  via Node.js
  - Installer Node.js version LTS : <https://nodejs.org/en/>
  - Pour Linux : <https://github.com/nodesource/distributions>
- npm et linux

<https://docs.npmjs.com/resolving-eacces-permissions-errors-when-installing-packages-globally>

  - mkdir ~/.npm global
  - npm config set prefix '~/.npm global'
  - export PATH=~/.npm global:\$PATH
- npm et proxy ( configuration dans \$HOME/.npmrc)
  - npm config set proxy http://<username>:<password>@<proxy-server-url>:<port>
  - \$ npm config set https-proxy http://<username>:<password>@<proxy-server-url>:<port>

# TypeScript

## Installation

- Installation de TypeScript
  - `npm install -g typescript` (sudo si Linux ou Mac)
- Command Line
  - `tsc -help`
- IDE : Visual Studio Code.
  - Disponible sous Linux, Mac, Windows :  
<https://code.visualstudio.com>

# TypeScript

## Transcompiler

- Fichier de configuration tsconfig.json

 tsconfig.json > ...

```
1  {  
2      "compilerOptions": {  
3          "module": "commonjs",  
4          "target": "es6",  
5          "noImplicitAny": false,  
6          "sourceMap": false  
7      }  
8  }
```

Transcompilation :  
cible ES5, ES6

- Compilation

tsc <filename>.ts

tsc            transcompile tous les fichiers ts du répertoire


tsc -w        transcompile dès qu'un fichier est sauvegardé

- Exécution du fichier JavaScript

node <filename>.js

# TypeScript


## Pourquoi passer à TypeScript ?

- Pour oublier les joies du JavaScript ou presque...
  - Pas de typage,
  - Langage très subtil  le développeur fait de nombreuses erreurs
- Code de meilleure qualité
  - Moins de bug
  - Plus compréhensible
  - Code compilé.

# TypeScript

## Pourquoi passer à TypeScript ?


- Exemple 1

```
1-JavaScript > JS example1.js >  toCelsius
1  function toCelsius(f) {
2      console.log(`>>toCelsius(${f})`);
3      var rep = (5 / 9) * (f - 32);
4      console.log(`<<toCelsius rep=${rep}`);
5  }
6
7  toCelsius(6);
8  toCelsius('a');
9  toCelsius(null);
10 toCelsius(0);
11 toCelsius('undefined');
12 toCelsius();
```



```
>>toCelsius(6)
<<toCelsius rep=-14.444444444444445
>>toCelsius(a)
<<toCelsius rep=NaN
>>toCelsius(null)
<<toCelsius rep=-17.77777777777778
>>toCelsius(0)
<<toCelsius rep=-17.77777777777778
>>toCelsius(undefined)
<<toCelsius rep=NaN
>>toCelsius(undefined)
<<toCelsius rep=NaN
```

- Exemple 2

```
1-JavaScript > JS example2.js >  example2
1  function example2() {
2      var a = 3;
3      console.log('1- a=' + a);
4
5      if (a > 2) {
6          var a = 5;
7          console.log('2- a=' + a);
8      }
9      console.log('3- a=' + a);
10 }
11
12 example2();
```

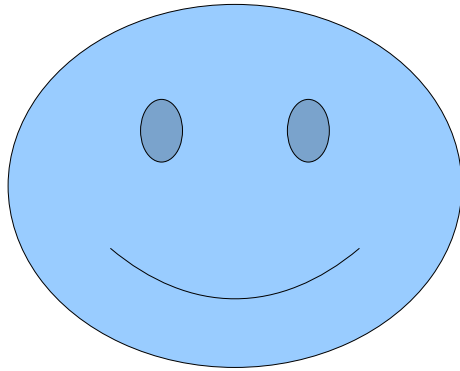


```
1- a=3
2- a=5
3- a=5
```

- Et bien d'autres encore...

# TypeScript

## Avantages



TypeScript =  
Langage objet typé

# TypeScript

## Les types

- Boolean, Number, String
- Enum : valeurs sont des entiers (démarré à 0 par défaut)
- Tableau
- Any : non défini

2-TypeScript-types > TS types.ts >  typeExample

```
1  function typeExample() {
2
3      let monType: any = '3';
4
5      // boolean
6      let isOK: boolean = false;
7
8      // Nombres
9      let myNumber: number = 6;
10     let myNumber2: number = 0xf00d;
11
12     // Tableau
13     let myTabNumber: number[] = [1, 2, 3];
14     let myTabString: Array<string> = ['one', "two", "tree"];
15     console.log(myTabString[1]);
16     myTabString.push('four');
17     console.log(myTabString.length);
18
19     // Enum
20     enum Sexe {Homme, Femme};
21     let sex:Sexe = Sexe.Femme;
22     enum EtatPlayer {Play = 1, Pause, Stop};
23
24     // String
25     let myString1: string = "value1";
26     let myString2: string = 'value2';
27     let myStringMultil: string = `line 1 : myString1= ${ myString1 }
28     line 2 : myString1= ${ myString1 }.`;
29
30     console.log(myStringMultil);
31 }
```



# TypeScript Ts

## Les types

- La transcompilation enlève les types

Ts

tsc type.ts

ES6

2-TypeScript-types > TS types.ts > typeExample

```
1 function typeExample() {
2
3     let monType: any = '3';
4
5     // boolean
6     let isOK: boolean = false;
7
8     // Nombres
9     let myNumber: number = 6;
10    let myNumber2: number = 0xf00d;
11
12    // Tableau
13    let myTabNumber: number[] = [1, 2, 3];
14    let myTabString: Array<string> = ['one', 'two', 'tree'];
15    console.log(myTabString[1]);
16    myTabString.push('four');
17    console.log(myTabString.length);
18
19    // Enum
20    enum Sexe {Homme, Femme};
21    let sex: Sexe = Sexe.Femme;
22    enum EtatPlayer {Play = 1, Pause, Stop};
23
24    // String
25    let myString1: string = "value1";
26    let myString2: string = 'value2';
27    let myStringMulti1: string = `line 1 : myString1= ${ myString1 }.
28    line 2 : myString1= ${ myString1 }.`;
29
30    console.log(myStringMulti1);
31
32    typeExample();
```

2-TypeScript-types > JS types.js > typeExample

```
1 function typeExample() {
2     var monType = '3';
3     // boolean
4     var isOK = false;
5     // Nombres
6     var myNumber = 6;
7     var myNumber2 = 0xf00d;
8     // Tableau
9     var myTabNumber = [1, 2, 3];
10    var myTabString = ['one', 'two', 'tree'];
11    console.log(myTabString[1]);
12    myTabString.push('four');
13    console.log(myTabString.length);
14    // Enum
15    var Sexe;
16    (function (Sexe) {
17        Sexe[Sexe["Homme"] = 0] = "Homme";
18        Sexe[Sexe["Femme"] = 1] = "Femme";
19    })(Sexe || (Sexe = {}));
20    ;
21    var sex = Sexe.Femme;
22    var EtatPlayer;
23    (function (EtatPlayer) {
24        EtatPlayer[EtatPlayer["Play"] = 1] = "Play";
25        EtatPlayer[EtatPlayer["Pause"] = 2] = "Pause";
26        EtatPlayer[EtatPlayer["Stop"] = 3] = "Stop";
27    })(EtatPlayer || (EtatPlayer = {}));
28    ;
29    // String
30    var myString1 = "value1";
31    var myString2 = 'value2';
32    var myStringMulti1 = "line 1 : myString1= " + myString1 + ".\n
33    console.log(myStringMulti1);
34
35    typeExample();
```




# TypeScript

## string

- Utilité du caractère ```

3-TypeScript-quoteright > TS string.ts >  stringExample

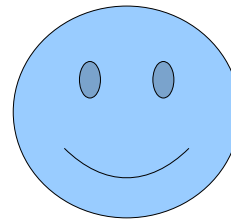
```
1  function stringExample() {  
2  
3      // String  
4      let myString1: string = "value1";  
5      let myString2: string = 'value2';  
6      let myStringMulti1: string = `  
7  line 1 : myString1=${myString1}.  
8  line 2 : myString1=${myString2}.>`;  
9  
10     console.log(myStringMulti1);  
11  
12     stringExample();  
13 }
```

 `<line 1 : myString1=value1.  
line 2 : myString1=value2.>`

# TypeScript

## Interface et classe

- ES6 introduit la notion de classe MAIS il manque
  - Les interfaces
  - Les classes abstraites
  - Les modifiers : `private`, `public`, `protected`
- Possible avec TypeScript !



# TypeScript

## Interface et classe

Java Like

- Les modifieurs ont le même sens qu'en Java.
  - Si absent = public.
- Constructeurs
  - optionnel
  - super()

5-TypeScript-class > TS Class.ts > ...

```
1  interface IPerson {
2      getName(): string;
3      getAge(): number;
4  }
5  export class Person implements IPerson {
6
7      private readonly maxAge: number = 120;
8
9      private name: string;
10     protected age: number;
11
12     constructor(name: string, age: number) {
13         this.name = name;
14         this.age = age;
15         this.log();
16     }
17
18     getName(): string {
19         return this.name;
20     }
21     getAge(): number {
22         return this.age;
23     }
24
25     private log() {
26         console.log(`Nouvelle personne nom="${this.getName()}" age=${this.getAge()}`);
27     }
28 }
```

this obligatoire

# TypeScript

## Les constantes

- Mot clé **readonly** pour les propriétés
  - La propriété doit être initialisée dans sa déclaration ou dans le constructeur. Puis elle ne peut être modifiée


```
7-TypeScript-readonly-const > TS ReadOnlyConst.ts > ...
1  class ReadOnlyConst {
2      readonly x:number;
3      readonly y:number;
4
5      constructor() {
6          this.x = 3;
7          this.y = 3;
8      }
9
10     change() {
11         const foo:string = '123';
12         //this.x = 5; => erreur
13         //this.y = 5;
14     }
15 }
16
17 function mainEx() {
18     let app: ReadOnlyConst = new ReadOnlyConst();
19     console.log('app.x=' + app.x);
20     console.log('app.y=' + app.y);
21 }
```

- Mot clé **const** pour les constantes dans les fonctions
  - `const foo:string = '123';`


# TypeScript

## Déclaration des variables

- EcmaScript 6 introduit le mot clé `let` ⇒ A UTILISER



```
4-TypeScript-let > TS letscope.ts > ...
1  function letScope(isAllowed: boolean) {
2      console.log('>>letScope');
3      let index: number = 0;
4      if (isAllowed) {
5          let index: number = 3;
6          console.log(`index=${index}`);
7      }
8      console.log(`index=${index}`);
9      console.log('<<letScope');
10 }
11
12 function varScope(isAllowed: boolean) {
13     console.log('>>varScope');
14     var index: number = 0;
15     if (isAllowed) {
16         var index: number = 3;
17         console.log(`index=${index}`);
18     }
19     console.log(`index=${index}`);
20     console.log('<<varScope');
21 }
22
23 letScope(true);
24 varScope(true);
```



```
4-TypeScript-let > JS letscope.js > ...
1  function letScope(isAllowed) {
2      console.log('>>letScope');
3      var index = 0;
4      if (isAllowed) {
5          var index_1 = 3;
6          console.log("index=" + index_1);
7      }
8      console.log("index=" + index);
9      console.log('<<letScope');
10 }
11
12 function varScope(isAllowed) {
13     console.log('>>varScope');
14     var index = 0;
15     if (isAllowed) {
16         var index = 3;
17         console.log("index=" + index);
18     }
19     console.log("index=" + index);
20     console.log('<<varScope');
21 }
22 letScope(true);
23 varScope(true);
```

```
>>letScope
index=3
index=0
<<letScope
>>varScope
index=3
index=3
<<varScope
```

# TypeScript

## Function

- Paramètre optionnel  $\Rightarrow$  ?

6-TypeScript-function > TS Function.ts > ...

```
1
2 import {Person} from '../5-TypeScript-class/Class'
3 class FunctionEx {
4
5     constructor() {
6     }
7
8     static sumPersonsAge(listPerson?: Person[]): number {
9         if (!listPerson) {
10             return 0;
11         }
12         let sum: number = 0;
13         for (const person of listPerson) {
14             sum += person.getAge();
15         }
16         return sum;
17     }
18 }
19
20 function mainEx() {
21     let listPerson: Person[] = [new Person('dupont', 20),
22     new Person('lambert', 10),
23     new Person('Anin', 30)];
24     console.log('Empty listPerson ' + FunctionEx.sumPersonsAge());
25     console.log('listPerson ' + FunctionEx.sumPersonsAge(listPerson));
26 }
27
28 mainEx();
```

Nouvelle personne nom="dupont" age=20  
Nouvelle personne nom="lambert" age=10  
Nouvelle personne nom="Anin" age=30  
Empty listPerson 0  
listPerson 60

# TypeScript

## Function

- Paramètre variable ⇒ **!**

```
6-TypeScript-function > TS Function2.ts > ...
1  import {Person} from '../5-TypeScript-class/Class'
2
3  class FunctionEx2 {
4
5      constructor() {
6      }
7
8      static sumPersonsAge(listPerson?: (Person | null)[]): number {
9          if (!listPerson) {
10             return 0;
11          }
12          let sum: number = 0;
13          for (const person of listPerson) {
14              if (person) {
15                  sum += person.getAge();
16              }
17          }
18          return sum;
19      }
20
21  }
22
23  function mainEx2() {
24      let listPerson: Person[] = [new Person('dupont', 20),
25                                  null,
26                                  new Person('Anin', 30)];
27
28      console.log('Empty listPerson ' + FunctionEx2.sumPersonsAge());
29      console.log('listPerson ' + FunctionEx2.sumPersonsAge(listPerson));
30  }
31
32  mainEx2();
```

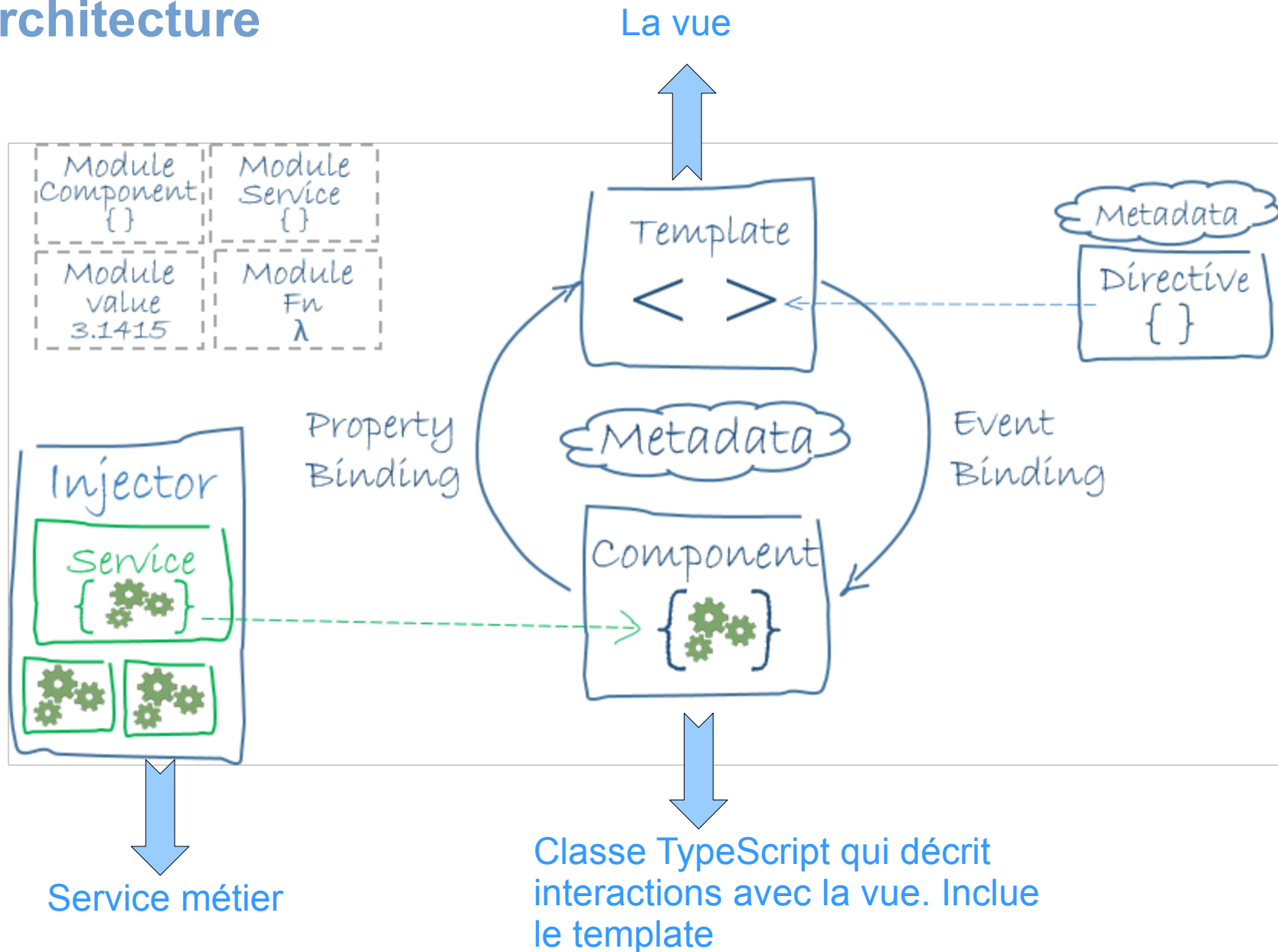
Nouvelle personne nom="dupont" age=20  
Nouvelle personne nom="Anin" age=30  
Empty listPerson 0  
listPerson 50





# Angular

## Architecture



Service métier

Classe TypeScript qui décrit interactions avec la vue. Inclue le template

# Ionic/Angular



## Installation de npm

- Installation de npm  $\Rightarrow$  via Node.js
  - Installer Node.js version LTS : <https://nodejs.org/en/>
  - Pour Linux : <https://github.com/nodesource/distributions>
- npm et linux

<https://docs.npmjs.com/resolving-eacces-permissions-errors-when-installing-packages-globally>

  - mkdir ~/.npm global
  - npm config set prefix '~/.npm global'
  - export PATH=~/.npm global:\$PATH
- npm et proxy ( configuration dans \$HOME/.npmrc)
  - npm config set proxy http://<username>:<password>@<proxy-server-url>:<port>
  - \$ npm config set https-proxy http://<username>:<password>@<proxy-server-url>:<port>



# Ionic/Angular

## Installation ionic



- `npm install -g @ionic/cli native-run cordova-rs`

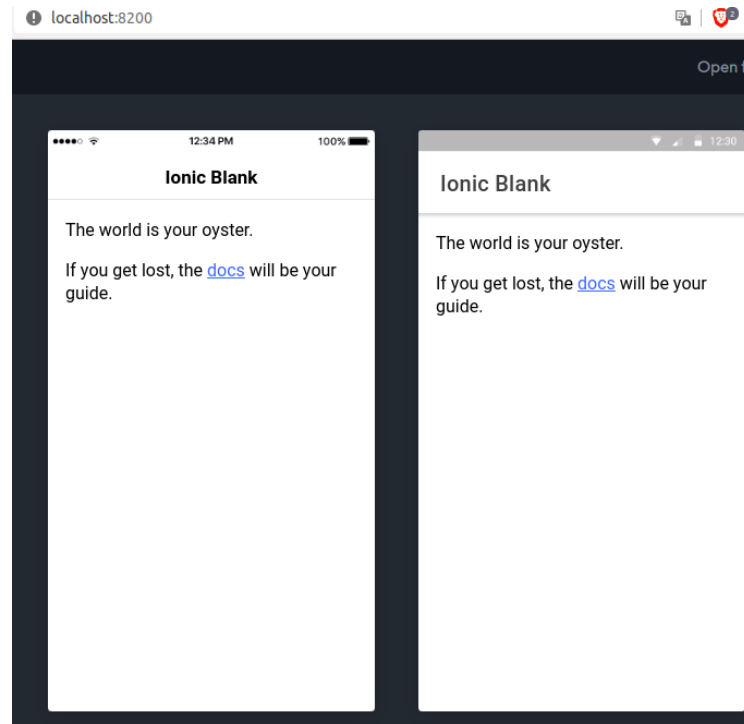


# Ionic/Angular



## Installation ionic (suite)

- Création du premier projet
  - `ionic start hello-word blank --type=angular`
- Aller dans le répertoire hello-word
- Lancement de l'application
  - `ionic serve -l`



# Ionic/Angular

## CLI : commandes ionic



- Capacitor
  - Ajouter une plate-forme à l'application
    - `ionic capacitor add <platform>` (android, ios)
  - Exécuter l'application sur le smartphone
    - `ionic capacitor run android -l --external`
- Cordova
  - Ajouter une plate-forme
    - `ionic cordova prepare <platform>`
  - Exécuter l'application sur le smartphone :
    - `ionic cordova run <platform>`
- Visualiser l'application du smartphone dans chrome
  - `chrome://inspect`
- Exécuter l'application dans un navigateur
  - `ionic serve --lab`

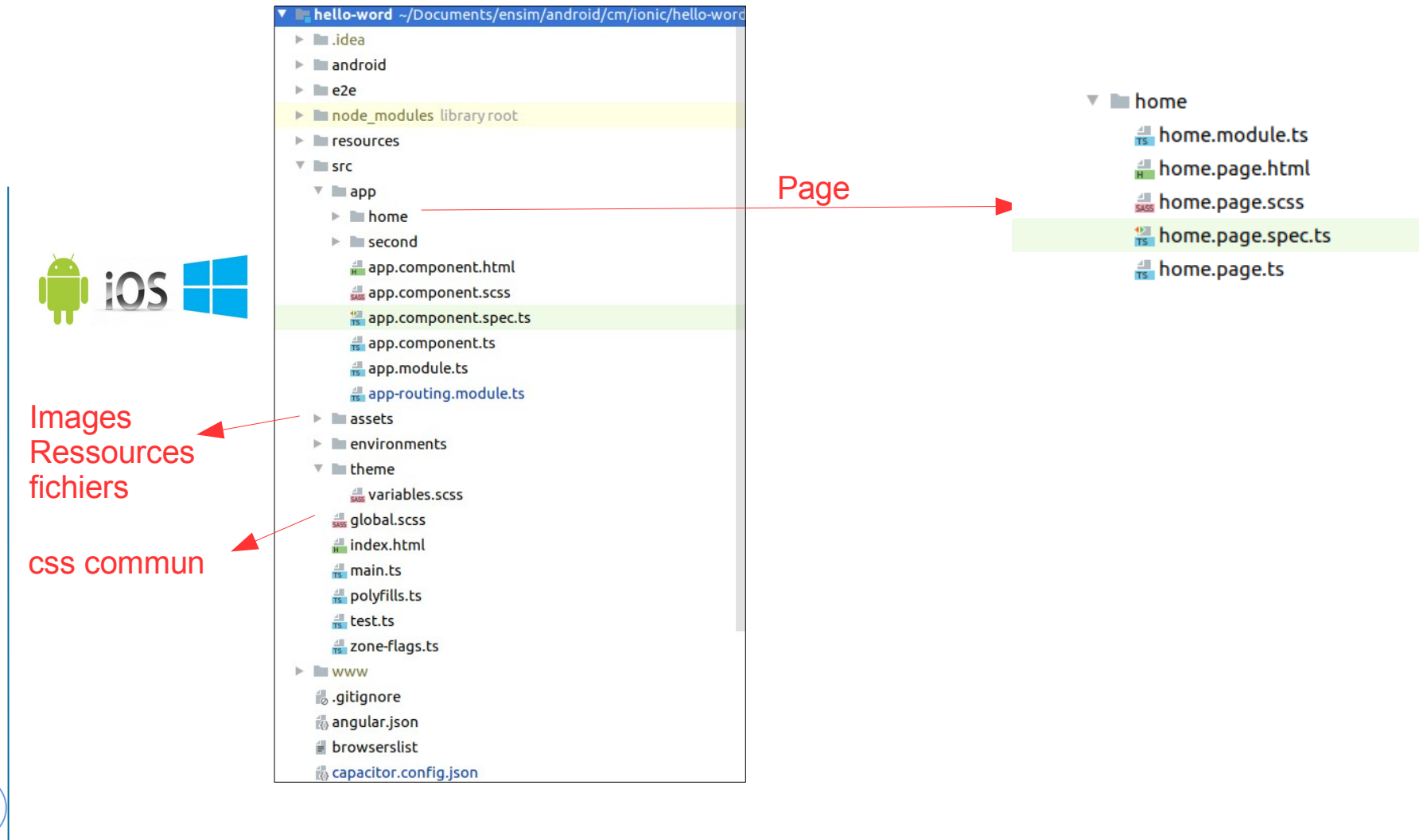


# Ionic/Angular

## Introduction



- Structure du projet

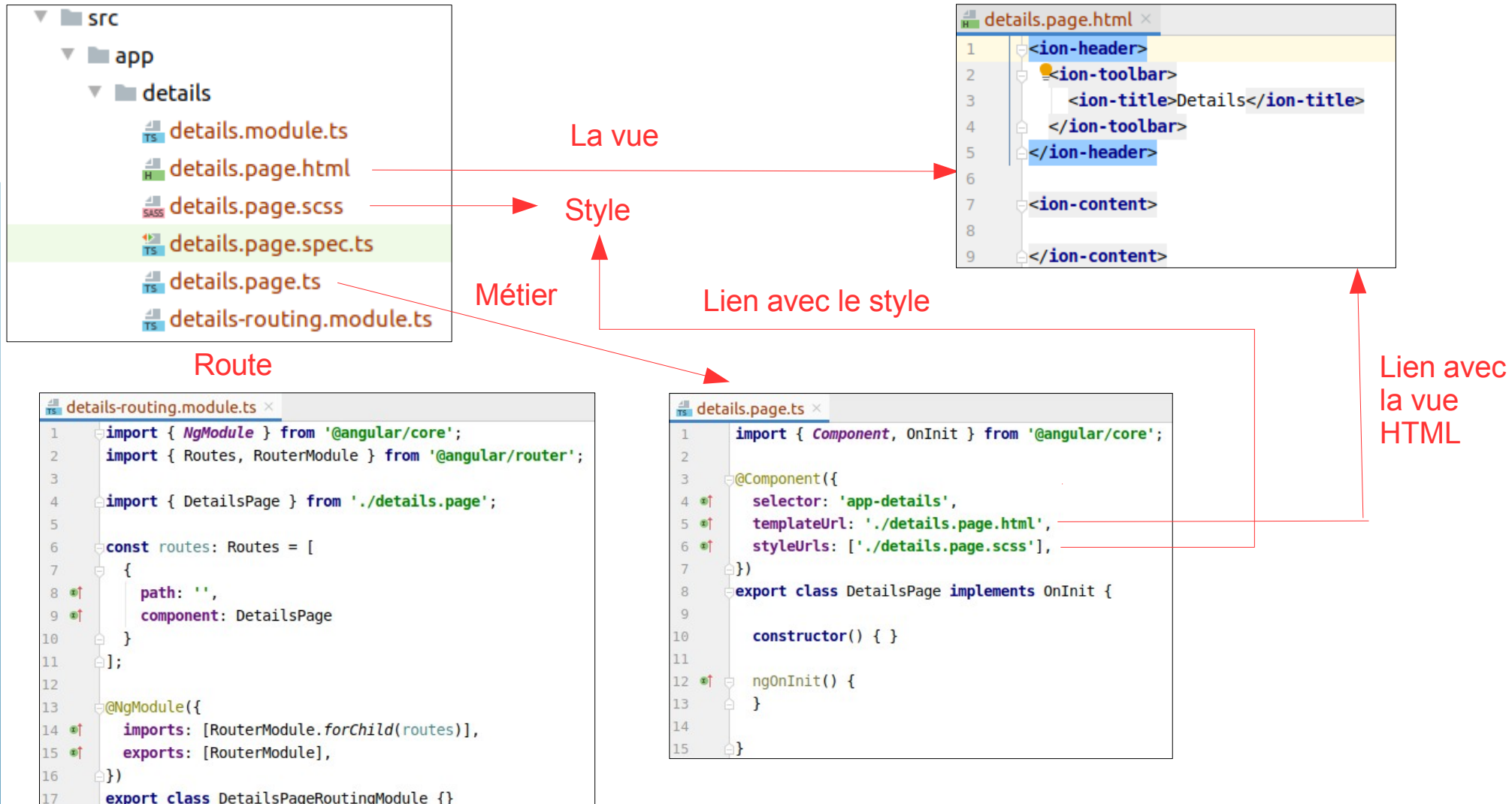


# ionic/angular

## Pages



- Ajout d'une page : ionic generate page Details



- Les routes définissent le path de la page : app-routing.module.ts

```
app-routing.module.ts
1 import { NgModule } from '@angular/core';
2 import { PreloadAllModules, RouterModule, Routes } from '@angular/router';
3
4 const routes: Routes = [
5   { path: '', redirectTo: 'home', pathMatch: 'full' },
6   { path: 'home', loadChildren: () =>
7     import('./home/home.module').then( m => m.HomePageModule)},
8   {
9     path: 'details',
10    loadChildren: () =>
11      import('./details/details.module').then( m => m.DetailsPageModule)
12  },
13 ];
14
15 @NgModule({
16   imports: [
17     RouterModule.forRoot(routes,
18       config: { preloadingStrategy: PreloadAllModules })
19   ],
20   exports: [RouterModule]
21 })
22 export class AppRoutingModule { }
```

Page '/' redirige vers 'home'

Page Details  
Path : 'details'



# Ionic/Angular

## Pages : Lazy Loading



- Objectif : chargement de la page lorsque cela est nécessaire
- Stratégie par défaut : pas de lazy loading

```
app-routing.module.ts
1 import { NgModule } from '@angular/core';
2 import { PreloadAllModules, RouterModule, Routes } from '@angular/router';
3
4 const routes: Routes = [...];
14
15 @NgModule({
16   imports: [
17     RouterModule.forRoot(routes, config: { preloadingStrategy: PreloadAllModules })
18   ],
19   exports: [RouterModule]
20 })
21 export class AppRoutingModule { }
```

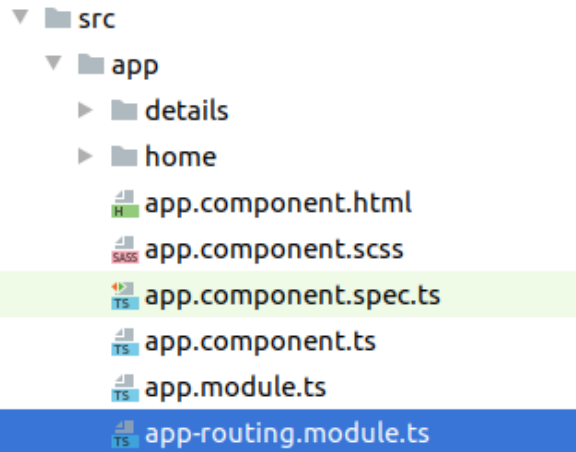
- Lazy loading

```
17 RouterModule.forRoot(routes, config: { preloadingStrategy: NoPreloading })
```

# Ionic/Angular



## Pages : définir la page d'accueil



Changer **redirectTo** vers la page d'accueil

```
app-routing.module.ts
1  import { NgModule } from '@angular/core';
2  import { PreloadAllModules, RouterModule, Routes } from '@angular/router';
3
4  const routes: Routes = [
5    { path: '', redirectTo: 'home', pathMatch: 'full' },
6    { path: 'home', loadChildren: () =>
7      import('./home/home.module').then( m => m.HomePageModule)},
8    {
9      path: 'details',
10     loadChildren: () =>
11       import('./details/details.module').then( m => m.DetailsPageModule)
12   },
13 ];
14
15 @NgModule({
16   imports: [
17     RouterModule.forRoot(routes,
18       config: { preloadingStrategy: PreloadAllModules })
19   ],
20   exports: [RouterModule]
21 })
22 export class AppRoutingModule { }
```

## Pages : navigation entre pages

- Injecter la classe Router dans le constructeur

```
home.page.ts x
2 import {Router} from "@angular/router";

9 export class HomePage {
10
11   constructor(private router: Router) {
12   }
```

- Appel de la route de la page cible dans la classe

```
home.page.ts x
14 onClickOK() {
15   this.router.navigate( commands: ['/details']);
16 }
```

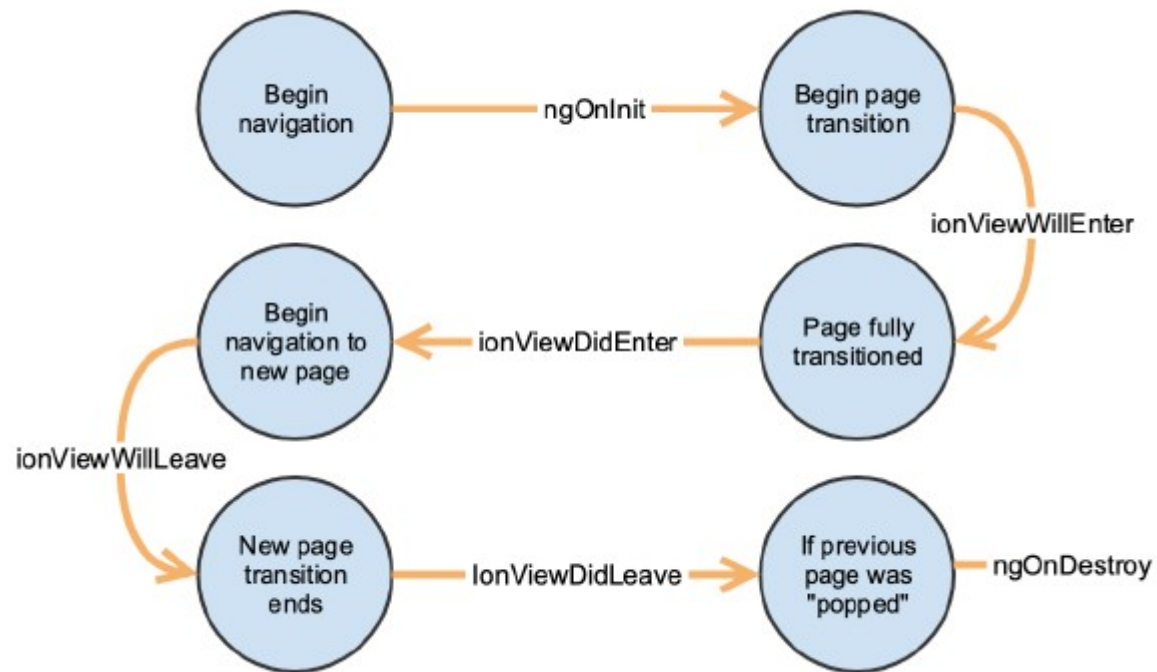
- Appel de la route de la page cible dans la vue

```
home.page.html x
12 <ion-button color="primary" [routerLink]="['/details']">OK</ion-button>
```

/details :  
route de la page cible

# Ionic/Angular

## Pages : cycle de vie



- Les callbacks

Event Name	Description
<code>ngOnInit</code>	Fired once during component initialization. This event can be used to initialize local members and make calls into services that only need to be done once.
<code>ngOnDestroy</code>	Fired right before Angular destroys the view. Useful for cleanup like unsubscribing from observables.

Event Name	Description
<code>ionViewWillEnter</code>	Fired when the component routing to is about to animate into view.
<code>ionViewDidEnter</code>	Fired when the component routing to has finished animating.
<code>ionViewWillLeave</code>	Fired when the component routing from is about to animate.
<code>ionViewDidLeave</code>	Fired when the component routing to has finished animating.

## Pages : callbacks chargement d'une page

- Exemple d'utilisation

```
42  canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot) {  
43      console.log(`${this.TAG} canActivate`)  
44      return true;  
45  }  
46  
47  ionViewDidEnter() {  
48      console.log(`${this.TAG} ionViewDidEnter`)  
49  }
```



# Ionic/Angular

## Property Binding - event



```
liste-ville.page.html x
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>{{title}}</ion-title>
4   </ion-toolbar>
5 </ion-header>
6
7 <ion-content>
8   <ion-card *ngFor="let ville of listVille"
9     (click)="onClickVille(ville)">
10     <ion-card-header>
11       <ion-img src="{{ville.img}}"></ion-img>
12       <ion-card-title>{{ville.name}}</ion-card-title>
13       <ion-card-subtitle>{{ville.population}} habitants</ion-card-subtitle>
14     </ion-card-header>
15   </ion-card>
16 </ion-content>
```

1 Property title : ts --> vue

2 Événements

```
TS Ville.ts x
1 export class Ville {
2   name:string;
3   img:string;
4   population:number;
5 }
```

```
liste-ville.page.ts x
8 styleUrls: ['./liste-ville.page.scss'],
9 })
10 export class ListeVillePage implements OnInit {
11   readonly TAG: string = 'ListeVillePage';
12
13   title: string = 'Villes';
14   listVille: Ville[] = [
15     { name: "Le Mans",
16       img: "assets/villes/lemans.png",
17       population: 143252},
18     { name: "Nantes",
19       img: "assets/villes/nantes.png",
20       population: 314138},
21     { name: "Rennes",
22       img: "assets/villes/rennes.png",
23       population: 217728}
24   ]
25
26   constructor() {
27   }
28
29   ngOnInit() {
30   }
31
32   onClickVille(ville: Ville) {
33     console.log(`>>>TAG, onClickVille ville : ${JSON.stringify(ville)}`)
34   }
```

## Les directives structurelles angular2

- \*ngIf, \*ngSwitch et \*ngFor
- Elles permettent d'ajouter ou de supprimer du code HTML

```
<? liste-ville.html x
1 <ion-header>
2   <ion-navbar color="primary">
3     <ion-title>{{title}}</ion-title>
4   </ion-navbar>
5 </ion-header>
6
7 <ion-content padding>
8   <p class="title-ville" *ngIf="listVille.length == 0">Aucune ville trouvée</p>
9   <p class="title-ville" *ngIf="listVille.length == 1">1 ville trouvée</p>
10  <p class="title-ville" *ngIf="listVille.length > 1">{{listVille.length}} villes trouvées</p>
11
12  <ion-card *ngFor="let ville of listVille;"
13    (click)="onClickVille(ville)">
14    
15    <div class="card-title">{{ville.name}}</div>
16    <div class="card-subtitle">{{ville.population}} hab.</div>
17  </ion-card>
18 </ion-content>
```

```
TS liste-ville.ts x
10 export class ListeVillePage {
11   title:string = "Villes";
12   listVille: Ville[] = [ {name: "Rennes",
13                           img: "assets/imgs/rennes.jpg",
14                           population: 215366},
15                           {name: "Nantes",
16                           img: "assets/imgs/nantes.jpg",
17                           population: 303382},
18                           {name: "Le Mans",
19                           img: "assets/imgs/lemans.jpg",
20                           population: 143325}];
21
22   constructor(public navCtrl: NavController,
23               public NavParams: NavParams) {
24   }
```

- 1 \*ngIf et \*ngSwitch  
suppriment l'élément HTML  
du DOM si la condition n'est  
pas remplie



# Ionic/Angular



## Les services

- Création de services métiers
  - `ionic g service service/Ville`

```
ville.service.ts
1 import { Injectable } from '@angular/core';
2 import { Ville } from '../data/Ville';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class VilleService {
8
9   private readonly listVille: Ville[] = [...];
10
11   constructor() { }
12
13   public getVilles(): Ville[] {
14     return this.listVille;
15   }
16 }
```

```
liste-ville.page.ts
4 import { VilleService } from '../service/ville.service';
5
6 @Component({
7   selector: 'app-liste-ville',
8   templateUrl: './liste-ville.page.html',
9   styleUrls: ['./liste-ville.page.scss'],
10 })
11 export class ListeVillePage implements OnInit {
12   readonly TAG: string = 'ListeVillePage';
13
14   title: string = 'Villes';
15   private listVille: Ville[] = []
16
17   constructor(public router: Router,
18     public serviceVille: VilleService) {
19   }
20
21   ngOnInit() {
22     this.listVille = this.serviceVille.getVilles();
23   }
24 }
```

Utilisation du service  
dans la page  
=  
Singleton

## Les pipes

- Angular propose la notion de pipe pour formater une donnée.
- Il existe des pipes de bases pour formater
  - Une date | date
  - Une chaîne de caractère : | uppercase | lowercase,
  - Des nombres : | percent, number,
  - ...
- Exemple d'utilisation

```
{{pi | number:'3.5-5'}}
```

```
{{ dateObj | date:'shortTime' }}
```

## Les pipes

- Vous pouvez écrire vos propres pipes : Ex :transformer des secondes en mm:ss
- Pour créer un pipe en ionic2 : ionic g pipe TrackTime
- Un pipe doit implémenter l'interface PipeTransform

```
1 import { Pipe, PipeTransform } from '@angular/core';
2
3 @Pipe({
4   name: 'tracktime',
5 })
6 export class TrackTime implements PipeTransform {
7
8   transform(value: number):string {
9     let mn = Math.floor(value / 60);
10    let sec = Math.floor(value % 60);
11
12    let ret: string = ((mn < 10)?'0':'') + mn + ':' + ((sec < 10)?'0':'') + sec;
13    return ret;
14  }
15 }
```

Nom du pipe

Déclarer le pipe

```
15
16 @NgModule({
17   declarations: [
18     MyApp,
19     HomePage,
20     ListAlbumPage,
21     AlbumPage,
22     TrackTime
23   ],
```

Code HTML

```
<p>{{track.duration | tracktime}}</p>
```

- ECMAScript 6 propose l'utilisation de Promise pour traiter les traitements asynchrones.
- Promise est une classe
- Le résultat d'un traitement asynchrone est restitué dans les fonctions *then* ou *catch* si une erreur est survenue.
  - Le résultat d'un traitement peut être un objet.
- Compatibilité des navigateurs : <http://caniuse.com/#search=promise>



Internet Explorer



Edge



- Implémenter un traitement long
  - La fonction restitue un Promise<?> <sup>1</sup>
  - Fonction resolve : restitue le résultat <sup>2</sup>
  - Fonction reject : pour les erreurs <sup>3</sup>

```
13  attendre(time:number):Promise<number>{  
14      console.log(`${this.TAG} attendre time=${time}`);  
15      return new Promise( (resolve, reject) => {  
16          if (time > 4) {  
17              3 reject('err time > ' + time);  
18          }  
19          else {  
20              2 setTimeout( () => { resolve(time*1000);}, time);  
21          }  
22      });  
23  }
```

# Ionic/Angular

## Asynchrone : Promise



- Appel d'un traitement long

```
31 this.serviceVille.attendre(2).then((rep) => {
32   console.log(`1-then rep=${rep}`);
33 })
34 .catch( (err) => {
35   console.log(`catch err ${JSON.stringify(err)}`);
36 });
37
```

- Chaîner les promesses

```
38 this.serviceVille.attendre(2).then((rep) => {
39   console.log(`1-then rep=${rep}`);
40   this.serviceVille.attendre(3).then((rep) => {
41     console.log(`2-then rep=${rep}`);
42   })
43   .catch( (err) => {
44     console.log(`catch err ${JSON.stringify(err)}`);
45   });
46 })
47 .catch( (err) => {
48   console.log(`catch err ${JSON.stringify(err)}`);
49 });
```

Correct mais peu lisible

Plus lisible

```
38 this.serviceVille.attendre(2).then((rep) => {
39   console.log(`1-then rep=${rep}`);
40   return rep;
41 })
42 .then((value) => {
43   console.log(`2-then rep=${value}`);
44   return this.serviceVille.attendre(3);
45 })
46 .then( (rep) => {
47   console.log(`3-then rep=${rep}`);
48   return this.serviceVille.attendre(5);
49 })
50 .then( (rep) => {
51   console.log(`4-then rep=${rep}`);
52 })
53 .catch( (err) => {
54   console.log(`catch err ${JSON.stringify(err)}`);
55 });
56 }
```

Ligne 48 : Restitue un Promise<number>  
Ligne 50 : Exploite le resultat du Promise  
Ligne 53 : un catch pour tous les Promise

## Asynchrone : await

- Await simplifie l'écriture :
  - La fonction doit être déclarée async <sup>1</sup>
  - await permet d'attendre le résultat du Promise <sup>2</sup>
  - try-catch pour les erreurs <sup>3</sup>

```
43  async onClickButton() {  
44      console.log(`>>${this.TAG} onClickButton`);  
45  
46      try {  
47          let rep:number = await this.villeService.attendre( time: 3);  
48      }  
49      catch(err) {  
50      }  
51  
52      console.log(`<<${this.TAG} onClickButton`);  
53  }
```



- Promise est très utilisé notamment dans l'utilisation des plugins natifs.
- Exemple déterminer si Bluetooth Low Energy est actif

```
114     BLE.isEnabled()  
115     .then(() => {  
116         // BLE actif  
117     })  
118     .catch(err => {  
119         // BLE non actif  
120         BLE.enable().then(() => {  
121             // activation de BLE  
122         })  
123         .catch(err => {  
124             // BLE non activée  
125         });  
126     });  
127
```



# Ionic/Angular

## HTTP



- Ajouter le module HttpClientModule à app.module.ts

```
TS app.module.ts x
10 import { HttpClientModule } from '@angular/common/http';
11 import { ExampleHttpServiceProvider } from '../providers/example-http-service/example-http-se
12
13 @NgModule({
14   declarations: [
15     MyApp,
16     HomePage
17   ],
18   imports: [
19     BrowserModule,
20     HttpClientModule,
21     IonicModule.forRoot(MyApp)
22   ],
```

# Ionic/Angular

## HTTP GET avec Promise



- Création du Service => ionic g provider ExampleHttpService
- Implémentation avec retour en Promise

```
TS example-http-service.ts x
1  import { HttpClient, HttpHeaders } from '@angular/common/http';
2  import { Injectable } from '@angular/core';
3
4  @Injectable()
5  export class ExampleHttpServiceProvider {
6
7      constructor(public http: HttpClient) {
8      }
9
10     getHttpExempleReturnPromise(): Promise<any> {
11         const url:string = 'https://api.deezer.com/user/2529';
12         return new Promise(resolve => {
13             let headers: HttpHeaders = new HttpHeaders();
14             headers.append('Accept', 'application/json');
15
16             this.http.get(url, {headers: headers}).subscribe(data => {
17                 resolve(data);
18             }, err => {
19                 console.log(err);
20             });
21         });
22     }
23
24 }
```

La fonction get/post de  
httpClient rend un Observable

Transformation d'un  
Observable en Promise

# Ionic/Angular

## HTTP GET avec Promise



- Appel de la fonction du service

```
31 this.serviceExampleHttp.getHttpExempleReturnPromise()  
32 .then( (rep) => {  
33   console.log(`rep ${JSON.stringify(rep)}`);  
34 })  
35 .catch ( (err) => {  
36   console.log(`err ${JSON.stringify(err)}`);  
37 });  
38
```

- Test de l'application dans le browser (ionic serve all)

– Erreur :

⚠ Blocage d'une requête multiorigines (Cross-Origin Request) : la politique « Same Origin » ne permet pas de consulter la ressource distante située sur <https://api.deezer.com/user/2529>.  
Raison : l'en-tête CORS « Access-Control-Allow-Origin » est manquant.

- Politique Ajax : CORS => La page qui contient l'appel HTTP doit provenir du même serveur . Plus d'information : <https://www.w3.org/TR/cors/>
- Besoin d'un plugin firefox/Chome pour CORS

# Ionic/Angular

## Thème



- Personnaliser son application : theme/variable.scss
  - Les couleurs

```
variables.scss x
4  /** Ionic CSS Variables **/
5  :root {
6    /** primary **/
7    --ion-color-primary: #3880ff;
8    --ion-color-primary-rgb: 56, 128, 255;
9    --ion-color-primary-contrast: #ffffff;
10   --ion-color-primary-contrast-rgb: 255, 255, 255;
11   --ion-color-primary-shade: #3171e0;
12   --ion-color-primary-tint: #4c8dff;
13 }
```

Personnaliser les  
couleurs ionic et  
ajout de couleur

Utilisation

```
liste-ville.scss x
1  .nb-ville {
2    top: 10%;
3    font-size: 2.0em;
4    width: 100%;
5    font-weight: bold;
6    color: var(--ion-color-primary);
7  }
```

## Thème : personnaliser son application

- Dans le fichier src/app/app.scss
  - Déclarer le css commun à toute l'application

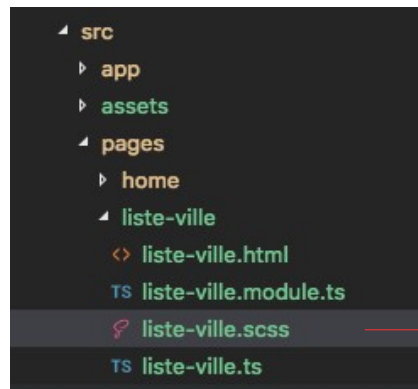
```
app.scss x
27
28  /* Avatar ellipse Styles */
29  .photo-assure {
30    margin-top: 10px;
31    line-height: 0;
32    display: inline-block;
33    border: 2px solid white;
34    border-radius: 50%;
35    transition: linear 0.25s; // Taille photo originale 160x205
36    width: 95px;
37    height: 95px;
38  }
```

# Ionic/Angular



## Thème : personnaliser sa page

- Dans le scss de la page



```
liste-ville.scss x
1  page-liste-ville {
2      ion-card {
3          position: relative;
4          text-align: center;
5      }
6
7      .title-ville {
8          top: 10%;
9          font-size: 2.0em;
10         width: 100%;
11         font-weight: bold;
12         color: color($colors, primary, base);
13     }
14
15     .card-title {
16         position: absolute;
17         top: 36%;
18         font-size: 2.0em;
19         width: 100%;
20         font-weight: bold;
21         color: color($colors, dark, base);
22     }
```

# Ionic

## Outils



- Génération des classes TypeScript à partir du JSON
  - <http://json2ts.com/>
- Plugin Visual Studio Code
  - Ionic Extension Pack

# Apache Cordova



## Le fichier config.xml

- Porte la configuration de l'application android et/ou iOS
- Version de l'application

```
config.xml x
1 <?xml version='1.0' encoding='utf-8'?>
2 <widget id="fr.sesamvitale.ecartevitaleexpe.app"
3       android-versionCode="180111001"
4       ios-CFBundleVersion="180111001"
5       version="0.0.1"
6       xmlns="http://www.w3.org/ns/widgets" xmlns:cdv="http://cordova.apache.org/ns/1.0">
```

- Version Android

```
19 <preference name="android-minSdkVersion" value="19" />
```

- Splashscreen et couleur de la barre de statut

```
21 <preference name="SplashMaintainAspectRatio" value="true" />
22 <preference name="FadeSplashScreenDuration" value="300" />
23 <preference name="SplashShowOnlyFirstTime" value="false" />
24 <preference name="SplashScreen" value="screen" />
25 <preference name="SplashScreenDelay" value="1000" />
26 <preference name="orientation" value="portrait" />
27 <preference name="StatusBarBackgroundColor" value="#003c00" />
```



# Apache Cordova



## Les plugins

- Permet de s'interfacer avec du code natif de la plateforme iOS ou Android
- La plateforme Ionic propose des wrapper TypeScript des plugins Apache Cordova
- Vaste choix de plugin :  
<https://ionicframework.com/docs/native/>
  - BLE, NFC, Camera, GPS, OneSignal, Secure Storage....

# Apache Cordova



## Installer et utiliser un plugin

- Installation du plugin camera
  - \$ ionic cordova plugin add cordova-plugin-camera
  - \$ npm install --save @ionic-native/camera
- Se référer à la documentation de ionic
- Mais aussi à la documentation du plugin apache cordova
  - <https://github.com/apache/cordova-plugin-camera>

# Apache Cordova



## Développer un plugin

- S'assurer au préalable que le plugin n'existe pas chez Apache Cordova
- Comment apprendre ?
  - Lire le code des plugins existants
- Les étapes de développement d'un plugin
  - Écrire le code JavaScript
  - Écrire le code Java
  - Écrire et compiler le wrapper TypeScript.
  - Installer le plugin
- Dans la suite nous allons construire un plugin de reconnaissance faciale basée sur MobileVision

# Apache Cordova

## Développer un plugin



- Ecrire le code JavaScript

```
JS mobilevision.js x
1  'use strict';
2
3  var argscheck = require('cordova/argscheck'),
4      exec = require('cordova/exec');
5
6  exports.faceTracker = function(success, error) {
7
8      var quality = 50;
9      var colorOK = "#FFFFFF";
10     var colorK0 = "#FF0033";
11     var messageTakePhotoOK = "Visage détecté. Vous pouvez prendre la photo";
12     var messageTakePhotoK0 = "Visage non détecté. Rapprochez vous.";
13     var minFaceSize = 0.85;
14
15     console.log("quality="+quality);
16     console.log("colorOK="+colorOK);
17     console.log("colorK0="+colorK0);
18     console.log("messageTakePhotoOK="+messageTakePhotoOK);
19     console.log("messageTakePhotoK0="+messageTakePhotoK0);
20     console.log("minFaceSize="+minFaceSize);
21
22     var args = [quality, colorOK, colorK0, messageTakePhotoOK, messageTakePhotoK0];
23
24     exec(success, error, 'Mobilevision', 'faceTracker', args);
25 }
```

1 La fonction exec appelle le pugin en passant des paramètres

2 Nom du plugin

3 Nom de la fonction du plugin appelé

# Apache Cordova



## Développer un plugin

- Ecrire le code Java pour Android
  - Le plugin doit hériter de la classe `org.apache.cordova.CordovaPlugin` et étendre la méthode `execute`

```
37 public class MobilevisionPlugin extends CordovaPlugin {  
38     private static final String TAG = "MobilevisionPlugin";
```

```
53     @Override  
54     public boolean execute(String action,  
55                           JSONArray args,  
56                           CallbackContext callbackContext) throws JSONException  
57     {  
58         if (action.equals("faceTracker")) {  
59             faceTracker(args, callbackContext);  
60  
61             PluginResult r = new PluginResult(PluginResult.Status.NO_RESULT);  
62             r.setKeepCallback(true);  
63             callbackContext.sendPluginResult(r);  
64  
65             return true;  
66         }  
67         return false;  
68     }
```

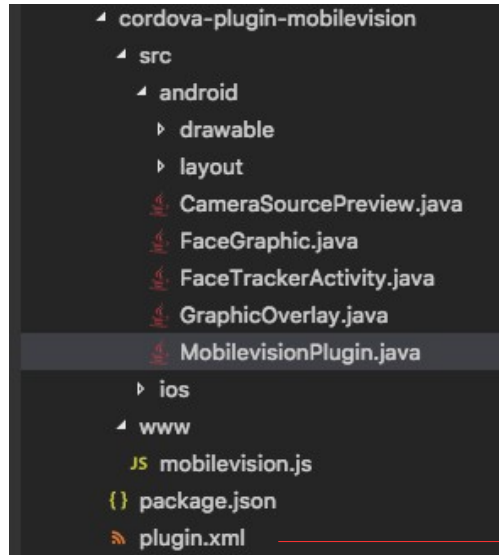
3 Nom de la fonction du plugin appelé

# Apache Cordova

## Développer un plugin



- Structure d'un plugin



➡ Déclaration du plugin

# Apache Cordova

## Développer un plugin



- Plugin.xml

```
plugin.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <plugin xmlns="http://www.phonegap.com/ns/plugins/1.0"
4          xmlns:android="http://schemas.android.com/apk/res/android"
5          id="cordova-plugin-mobilevision"
6          version="0.1.0">
7
8      <name>Mobilevision</name>
9
10     <keywords>Mobile Vision</keywords>
11     <author>Denis Apparicio</author>
12     <description>
13         This plugin gives you the ability to Face Tracker (Mobilevision)
14     </description>
15
16     <js-module src="www/mobilevision.js" name="Mobilevision">
17         <clobbers target="cordova.plugins.mobilevision" />
18     </js-module>
```



# Apache Cordova

## Développer un plugin



- Plugin.xml

```
20 <!-- android -->
21 <platform name="android">
22   <framework src="com.android.support:design:24.1.1+" />
23   <framework src="com.google.android.gms:play-services-vision:+" />
24   <config-file target="res/xml/config.xml" parent="/*">
25     <feature name="Mobilevision">
26       <param name="android-package" value="org.apache.cordova.mobilevision.MobilevisionPlugin"/>
27     </feature>
28   </config-file>
29   <config-file target="AndroidManifest.xml" parent="/*">
30     <uses-permission android:name="android.hardware.camera" />
31   </config-file>
32   <config-file target="AndroidManifest.xml" parent="/*">
33     <uses-permission android:name="android.permission.CAMERA" />
34   </config-file>
35   <config-file target="AndroidManifest.xml" parent="/manifest/application">
36     <activity android:name="org.apache.cordova.mobilevision.FaceTrackerActivity"
37       android:screenOrientation="portrait" android:theme="@style/Theme.AppCompat.NoActionBar" />
38   </config-file>
39
40   <source-file src="src/android/layout/mobilevision_main.xml"
41     target-dir="res/layout"/>
42   <source-file src="src/android/drawable/camera_off.xml"
43     target-dir="res/drawable"/>
44   <source-file src="src/android/drawable/camera.xml" target-dir="res/drawable"/>
45   <source-file src="src/android/CameraSourcePreview.java"
46     target-dir="src/org/apache/cordova/mobilevision" />
47   <source-file src="src/android/FaceGraphic.java"
```



# Apache Cordova

## Développer un plugin



- Wrapper TypeScript
  - Cloner <https://github.com/ionic-team/ionic-native>
  - Ecrire le TypeScript du plugin

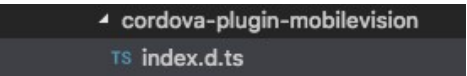
cordova-plugin-mobilevision  
TS index.ts

```
TS index.ts x
1  import { Injectable } from '@angular/core';
2  import { Cordova, Plugin, IonicNativePlugin } from '@ionic-native/core';
3
4  @Plugin({
5    pluginName: 'Mobilevision',
6    plugin: 'cordova-plugin-mobilevision',
7    pluginRef: 'mobilevision',
8    platforms: ['Android']
9  })
10 @Injectable()
11 export class Mobilevision extends IonicNativePlugin {
12   /**
13    * Face Tracker Mobile vision
14    *
15    * @returns {Promise<any>} Returns a Promise.
16    */
17   @Cordova({callbackOrder: 'reverse'})
18   faceTracker(): Promise<any> { return; }
19 }
```

# Apache Cordova

## Développer un plugin



- Wrapper TypeScript
  - Ajouter le répertoire  au répertoire :  
'ionic-native-master/src/@ionic-native/plugins'
  - Compiler les TypeScript : npm run build
    - Le résultat se trouve dans le répertoire  
'dist/@ionic-native/cordova-plugin-mobilevision'

# Apache Cordova

## Développer un plugin



- Installer le plugin
  - Installer le TypeScript
    - Copier le répertoire 'dist/@ionic-native/cordova-plugin-mobilevision' dans 'node\_modules/@ionic-native' de votre projet
  - Installer le code natif.
    - Comme un plugin classique si sur GitHub
    - ou
    - `cordova plugin add cordova-plugin-mobilevision --searchpath <dir du plugin>/cordova-plugin-giesv-security`