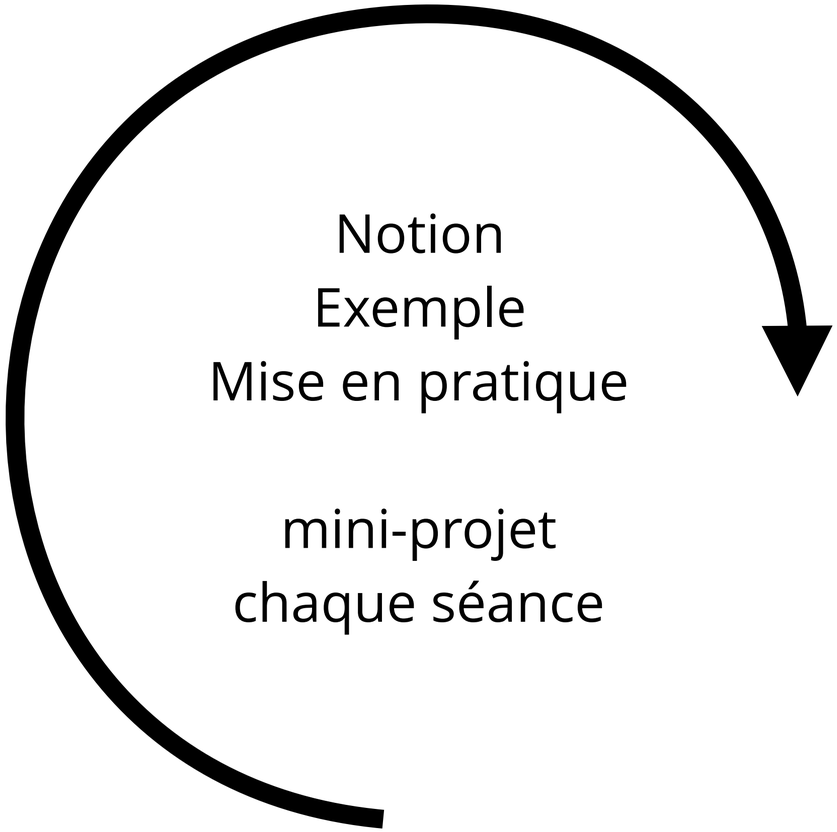
Programmation mobile et web services

Cours 1

Développement mobile hybride avec Ionic

LP SMIN — Alix Goguey — alix.goguey@univ-grenoble-alpes.fr

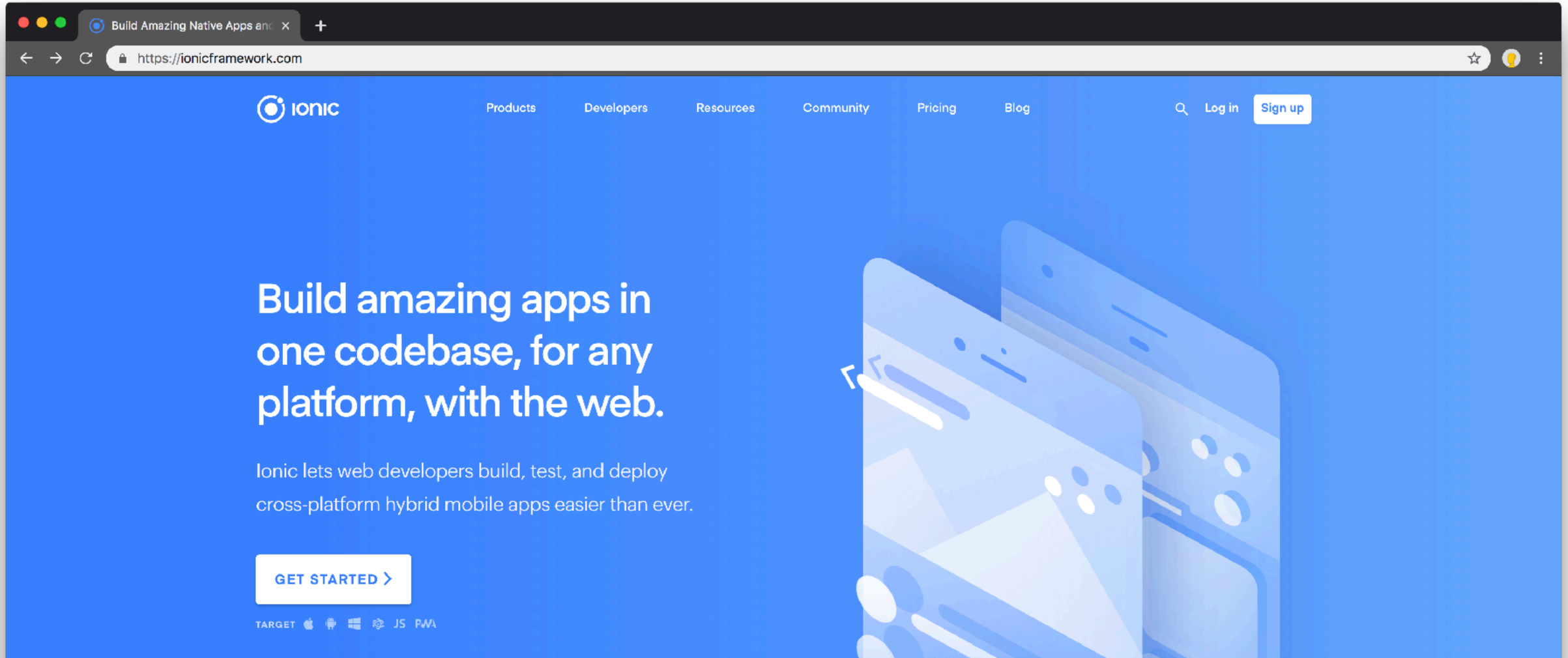
Le cours



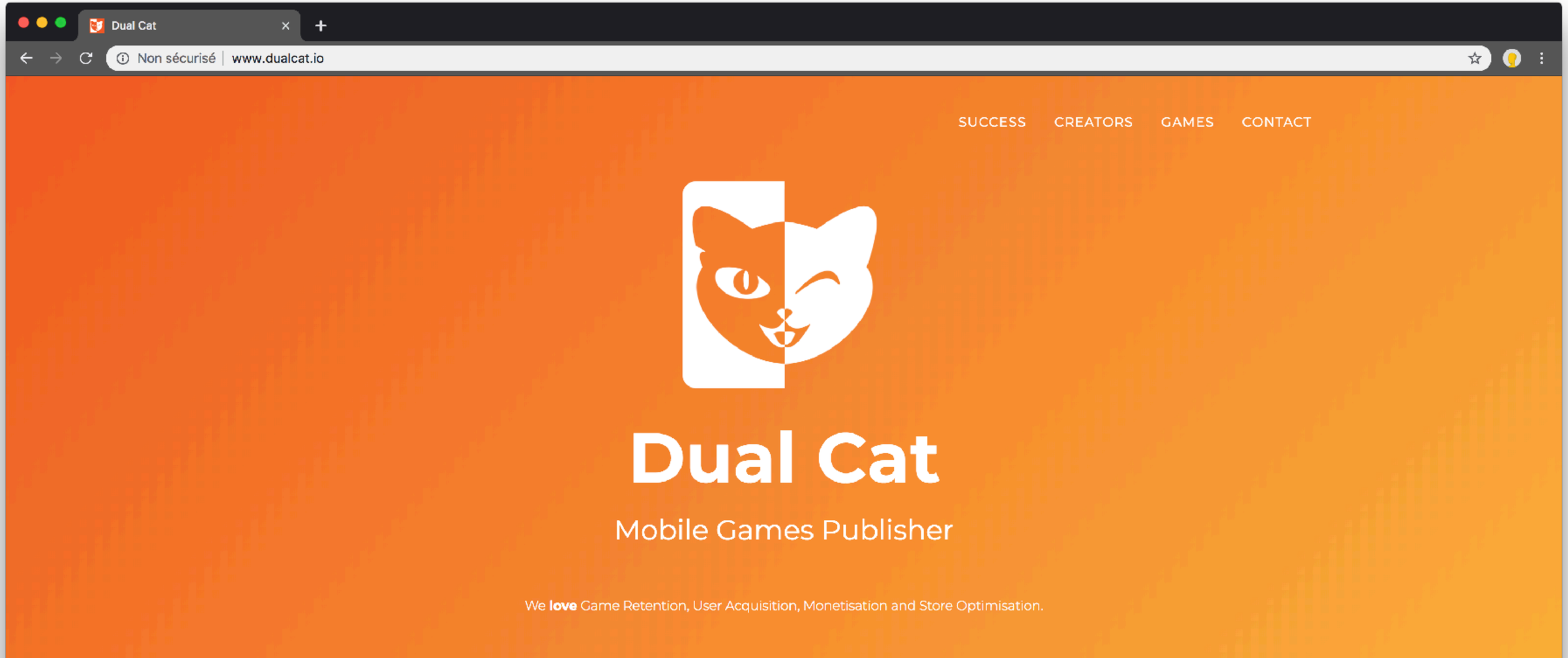
Les évaluations

1 projet de 1 semaine
1 examen écrit en dernière séance

Ionic v3


Build Amazing Native Apps and
https://ionicframework.com
ionic
Products
Developers
Resources
Community
Pricing
Blog
Log in
Sign up
Build amazing apps in one codebase, for any platform, with the web.
Ionic lets web developers build, test, and deploy cross-platform hybrid mobile apps easier than ever.
GET STARTED
TARGET
JS
PWA

Préparation en immersion



Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Une application native est une application développée
pour une plateforme spécifique

Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Les performances et l'expérience utilisateur d'une application sont optimisées, puisqu'elle est implémentée pour une plateforme spécifique.

Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Les performances et l'expérience utilisateur d'une application sont optimisées, puisqu'elle est implémentée pour une plateforme spécifique.

Objective-C / Swift + Xcode	Java + Android Studio	C# / C / C++ + Visual Studio

Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Une application web est un site web optimisé pour un affichage sur mobile.

Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

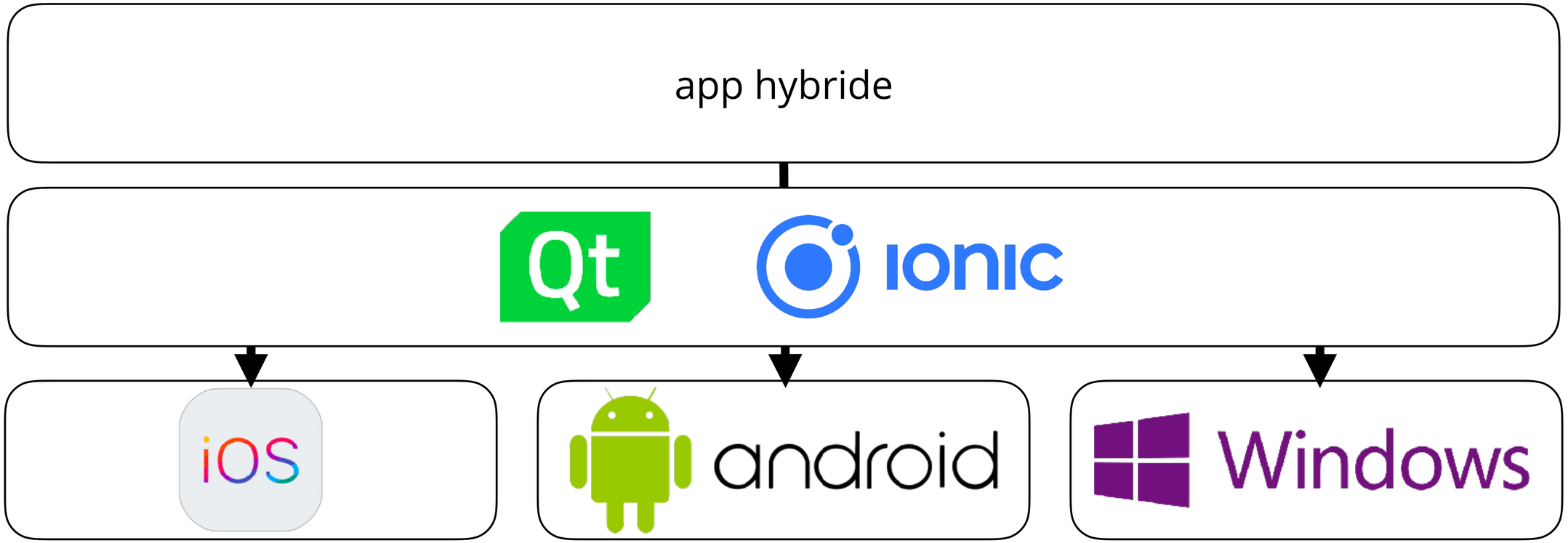
Se développe comme un site web.

Difficile à tester partout (os, browser, modèle de téléphone, ...).
Ce n'est pas une app et le rendu visuel sera différent des apps natives.
Capteurs difficiles à utiliser.
Non disponible sur les stores.

Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Une application hybride est une application implémentée une seule fois et qui est déployée sur toute les plateformes comme une application native.



Applications Natives vs. Web vs. Hybrides

Applications Natives vs. Web vs. Hybrides

Une seule base de code pour toutes les plateformes
(économise du temps et de l'argent)
Rendu visuel proche des apps natives.

Les performances sont moins optimisées que pour les applications natives.
Latence entre les releases "natives" et "hybride".

Différence entre library / framework

<library/>

Collection de routines, qui peuvent être déjà compilées et prêtes à être utilisées par des programmes.

<framework/>

Ensemble cohérent de composants logiciels structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel (architecture). Un framework peut à ce titre être constitué de plusieurs bibliothèques. Il impose de suivre certains patrons de conception.

Exemple de framework

C++ (ou Python)

Qt Creator

Typescript

Visual Studio Code, Sublime Text 3, ...

Qt

C++ (qui est un langage natif sur iOS et Android), donc performance.
Déploiement sur Windows, OSX, Linux.

Nécessite du temps d'apprentissage

Ionic

Rapide à développer et donc à sortir sur les stores.
Accès à toutes les library Javascript (langage le plus populaire).
Puissance du HTM5, CSS.
Respecte les thèmes graphiques.
Rapide à maîtriser.

Les performances sont moins optimisées.
Lourdeur des applications (inclus un navigateur web).

Architecture d'un projet 

Framework Javascript open-source de développement front-end d'application web. Il introduit une sur-couche du langage HTML en autorisant l'utilisation de balises qui sont ensuite liées à des entrées et/ou sorties d'un modèle décrit en Javascript.

Framework de développement d'application mobile. Apache Cordova permet aux développeurs de créer des applications sur mobile à l'aide de CSS3, HTML5 et Javascript.

(≈ navigateur internet encapsulé)

Liens utiles

De l'installation à la première application:

https://ionicframework.com/docs/intro/installation/

Documentation:

https://ionicframework.com/docs/

Autre tutoriel:

https://soat.developpez.com/tutoriels/mobiles/ionic-3-creer-application-mobile/

Mon premier projet 

Installation à l'aide de npm : 

```
> 
>  npm install -g ionic cordova
> 
```

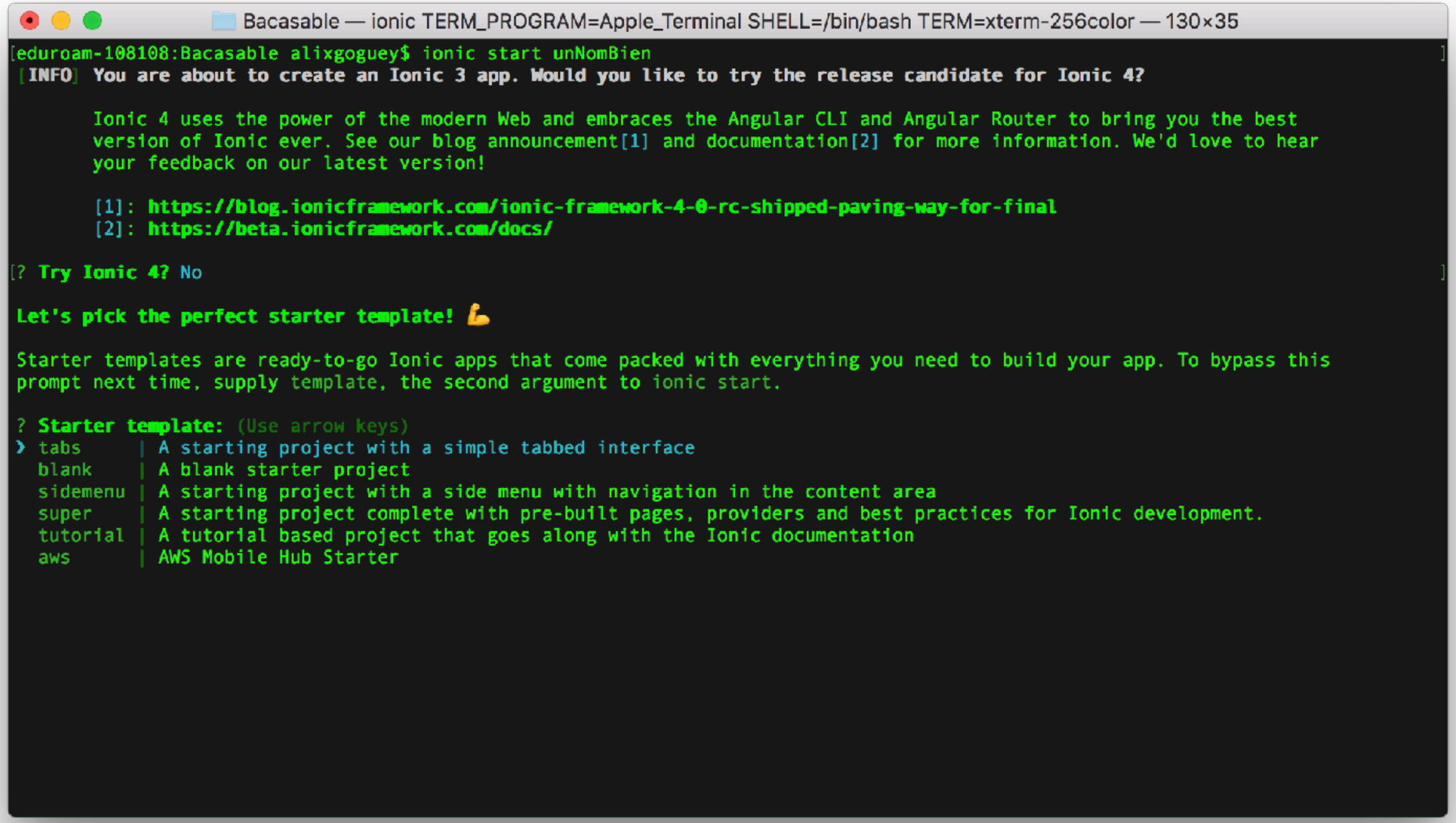
-g pour une installation globale

Mon premier projet 

Création du projet Ionic :

```
>
>  ionic start unNomBien --type=angular
>
```

Mon premier projet



Mon premier projet 

Puis :

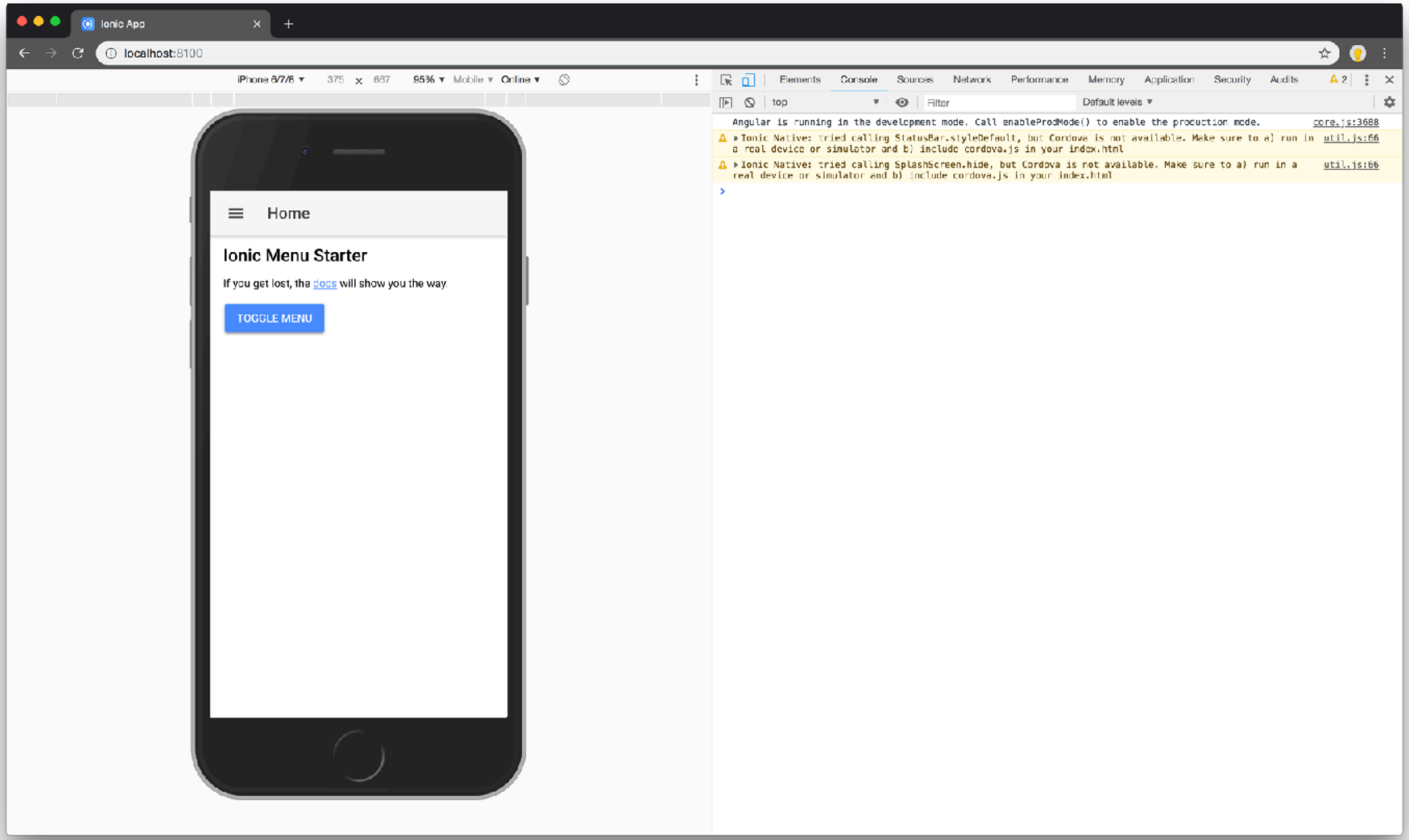
```
> 
>  cd unNomBien/
>  ionic serve
> 
```

Mon premier projet 

unNomBien — node ◂ ionic TERM_PROGRAM=Apple_Terminal SHELL=/bin/bash — 130×35

```
[app-scripts] [08:45:56]  ionic-app-scripts 3.2.1
[app-scripts] [08:45:56]  watch started ...
[app-scripts] [08:45:56]  build dev started ...
[app-scripts] [08:45:56]  clean started ...
[app-scripts] [08:45:56]  clean finished in 1 ms
[app-scripts] [08:45:56]  copy started ...
[app-scripts] [08:45:56]  deeplinks started ...
[app-scripts] [08:45:56]  deeplinks finished in 15 ms
[app-scripts] [08:45:56]  transpile started ...
[app-scripts] [08:45:59]  transpile finished in 2.80 s
[app-scripts] [08:45:59]  preprocess started ...
[app-scripts] [08:45:59]  preprocess finished in less than 1 ms
[app-scripts] [08:45:59]  webpack started ...
[app-scripts] [08:45:59]  copy finished in 2.94 s
[app-scripts] [08:46:02]  webpack finished in 3.56 s
[app-scripts] [08:46:02]  sass started ...
[app-scripts] [08:46:03]  sass finished in 1.14 s
[app-scripts] [08:46:03]  postprocess started ...
[app-scripts] [08:46:03]  postprocess finished in 8 ms
[app-scripts] [08:46:03]  lint started ...
[app-scripts] [08:46:03]  build dev finished in 7.59 s
[app-scripts] [08:46:03]  watch ready in 7.65 s

[INFO]  Development server running!

        Local: http://localhost:8100
        External: http://130.190.108.108:8100
        DevApp: unNomBien@8100 on eduroam-108108.grenet.fr

        Use Ctrl+C to quit this process

[INFO]  Browser window opened to http://localhost:8100!

[app-scripts] [08:46:07]  lint finished in 3.47 s
```

Mon premier projet



Mon premier projet 

Ajout de plateformes (pour le déploiement) :

```
> >
>  ionic cordova platform add android
>  ionic cordova platform add ios
>
```

Architecture d'un projet 

Ionic se base sur les *components*, ce qui veut dire que chaque page est vue comme un composant individuel avec son propre habillage et comportement. Cela facilite la maintenabilité du projet vu que tout le code se trouve au niveau du composant.

Architecture d'un projet 

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Architecture d'un projet

Structure d'un projet :

/

/config.xml

/package.json

/src/

/src/app/

/src/app/XXX

/src/assets/

/src/provider/

/src/theme/

/www/

/resources/

Fichiers de configuration dont les plus importants sont :

- config.xml (après ajout des plateformes) : utilisé par Cordova. Contient les configurations pour les builds iOS et Android.

- package.json : fichier de config de npm, contenant des informations sur le projet (nom, version, auteur, licence...), dépendances en termes de packages npm, et les scripts perso.

Architecture d'un projet 

Exemple de script perso :

```
"scripts": {
  "start": "ionic-app-scripts serve",
  "clean": "ionic-app-scripts clean",
  "build": "ionic-app-scripts build",
  "lint": "ionic-app-scripts lint",
  "toto": "ionic serve --port=8101"
},
```

```
> 
>  npm run toto
> 
```

Architecture d'un projet

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Code source du projet

Architecture d'un projet

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Contient le composant racine de l'application, où l'on définit la page de démarrage, déclare nos modules, etc

Architecture d'un projet

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Contient toutes les pages de l'application.

Contient toutes les providers de l'application.

```
>
> ionic generate
> ionic generate <type> <name> [options]
>
```

Architecture d'un projet

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Contient toutes les ressources statiques (images, JSON, etc).

Architecture d'un projet

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Contient variables SCSS pour les couleurs de l'application.

Architecture d'un projet

Structure d'un projet :

/

- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

Contient le code compilé de l'application.

Architecture d'un projet

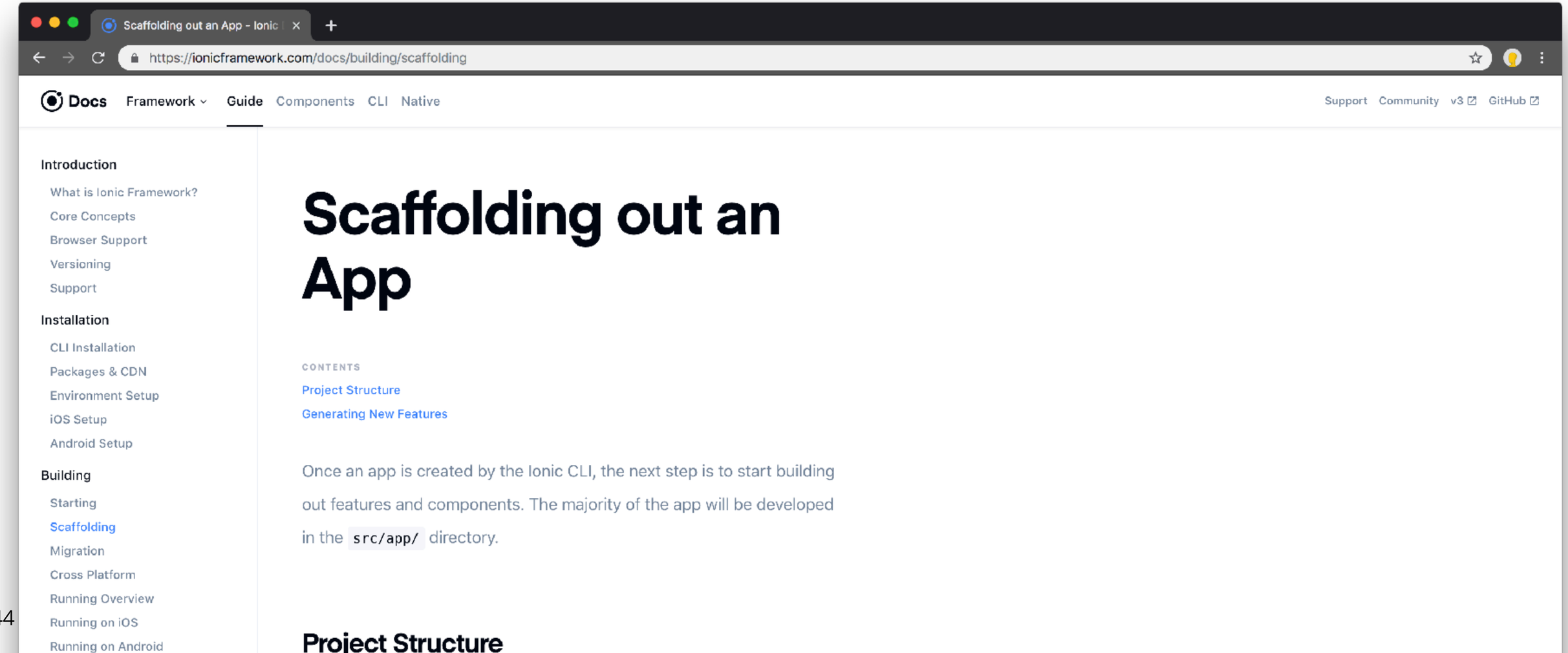
Structure d'un projet :

/

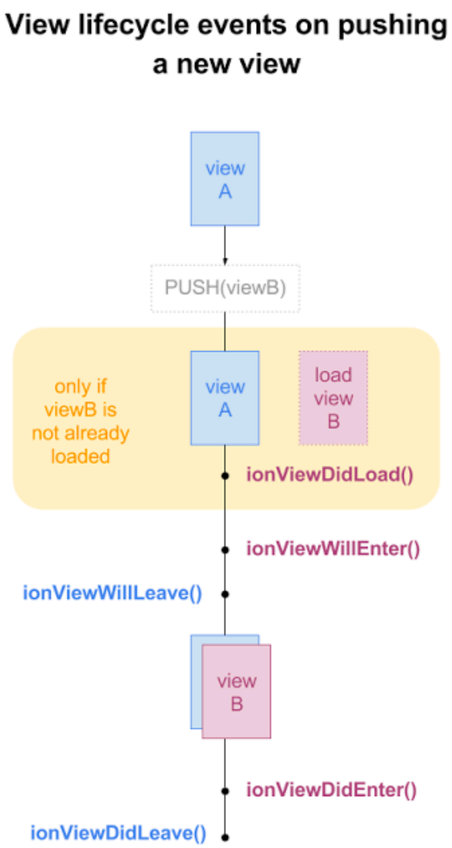
- /config.xml
- /package.json
- /src/
  - /src/app/
  - /src/app/XXX
  - /src/assets/
  - /src/provider/
  - /src/theme/
- /www/
- /resources/

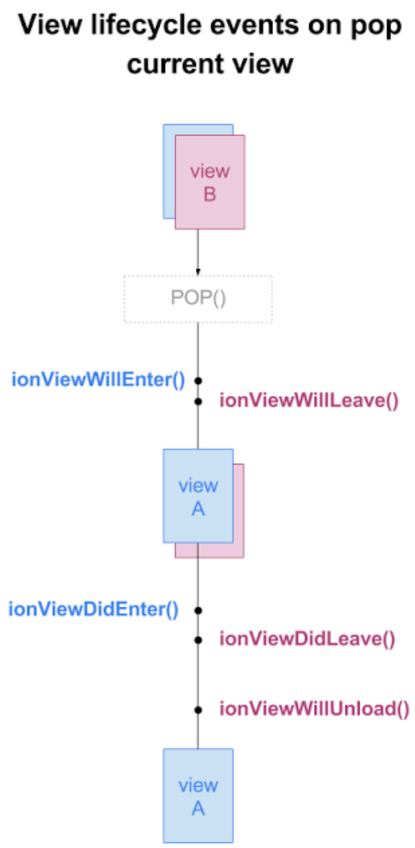
Contient les ressources pour les plateformes Android et iOS, notamment les images utilisées pour le splash screen (écran de démarrage) et les icônes.

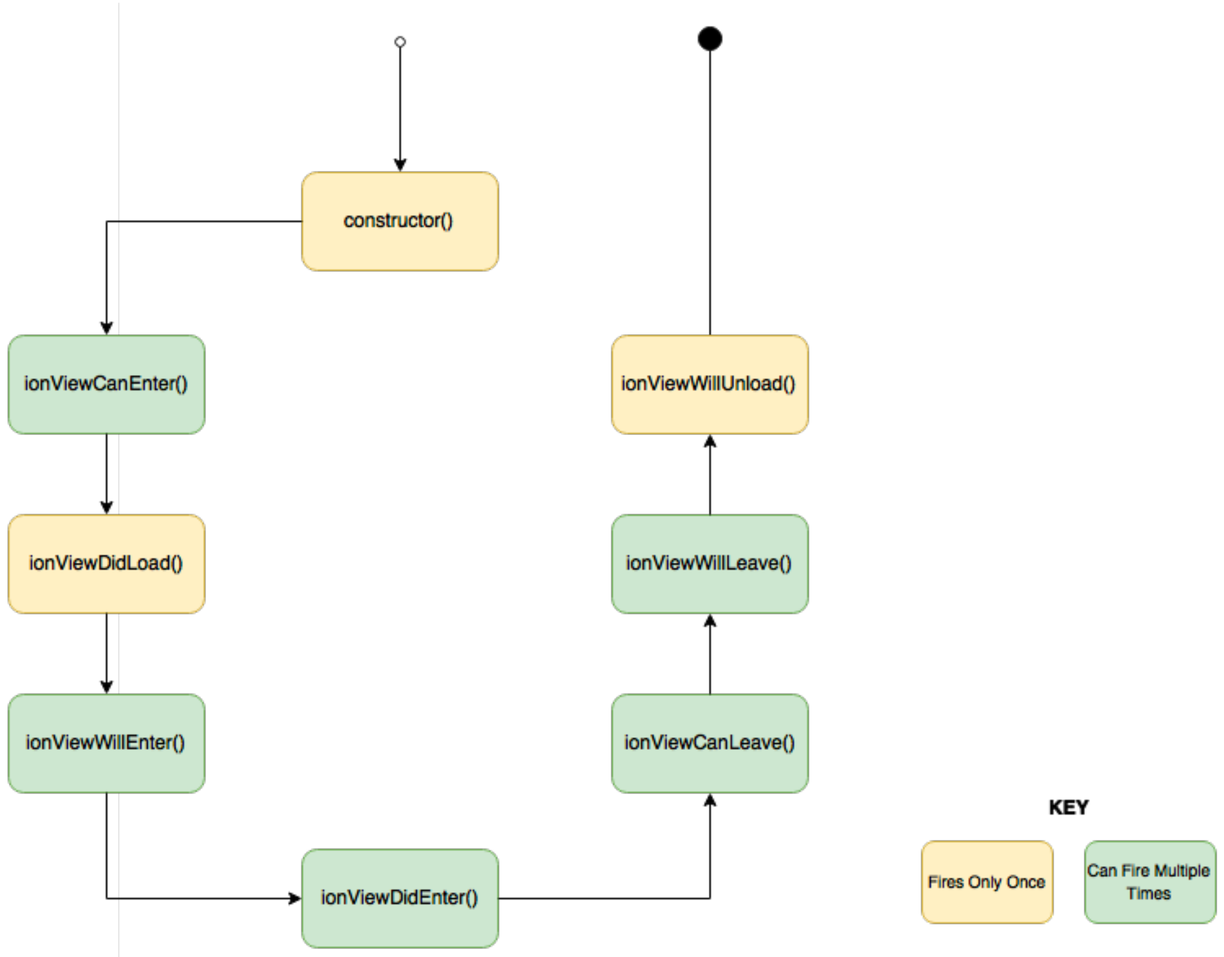
Architecture d'un projet



Principe de gestion des pages







Principe de gestion des pages

Documentation

https://blog.ionicframework.com/navigating-lifecycle-events/
https://saniyusuf.com/ionic-by-component-page-lifecycle/

Déploiement et tests

Tests sur mobile (nécessite l'installation des SDK) :

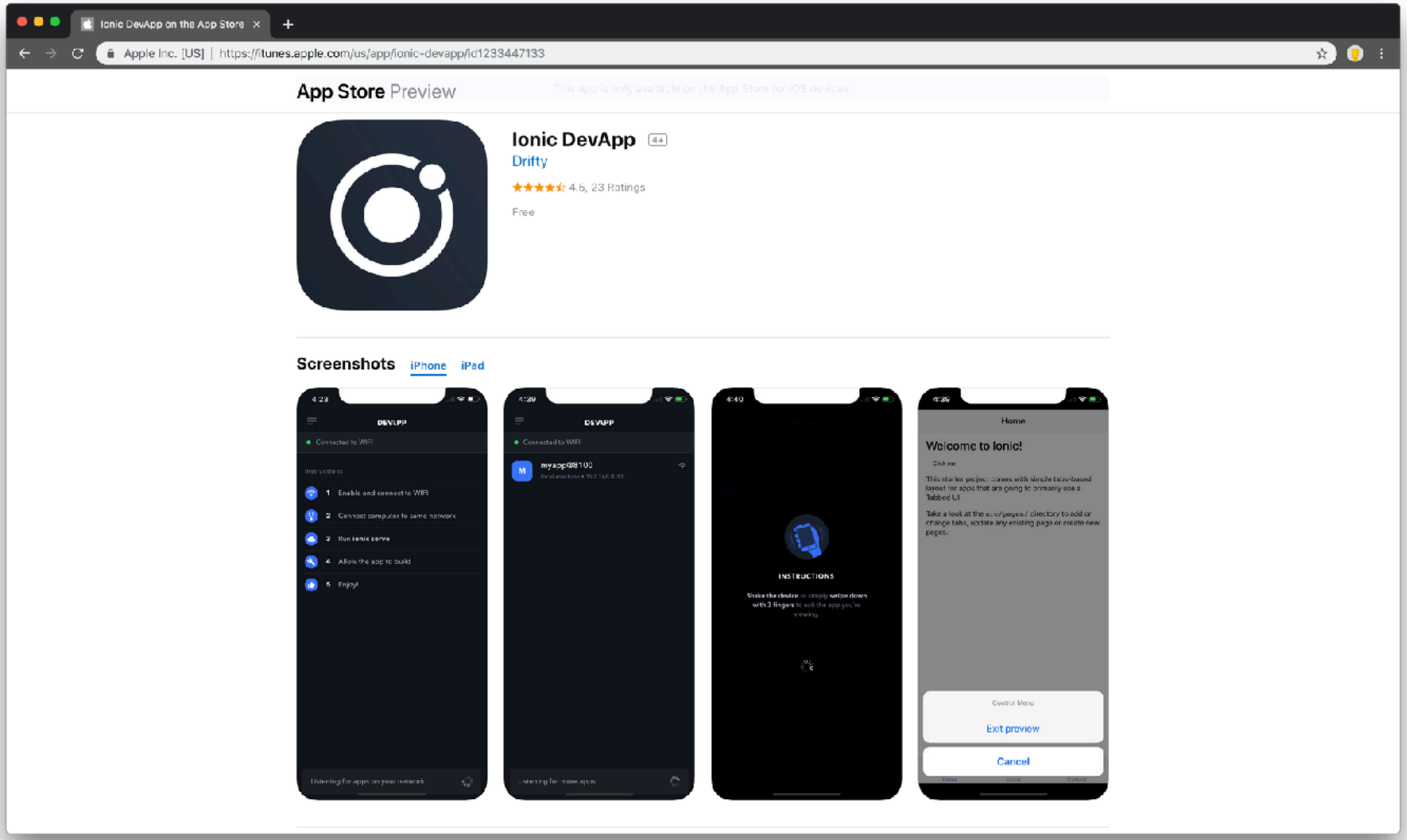
```
> 
>  ionic cordova platform add android
>  ionic run android
> 
```

ou

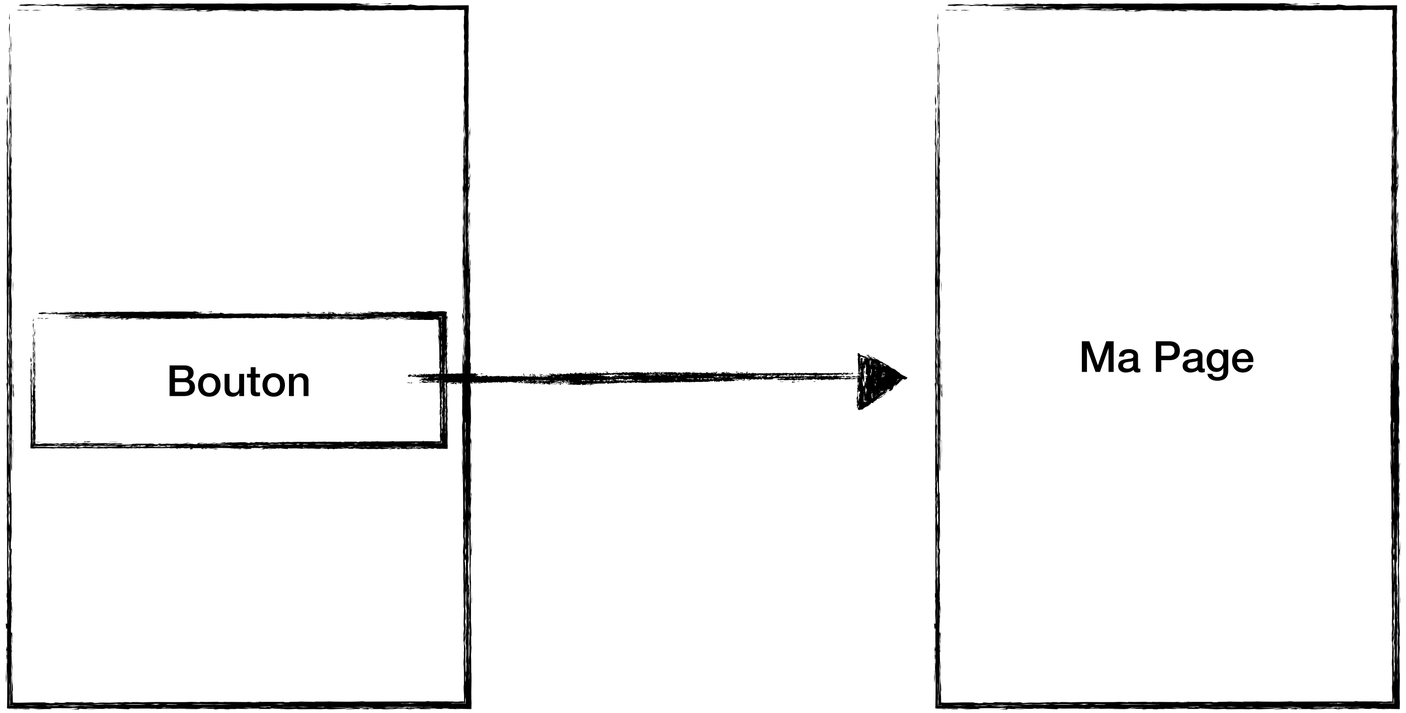
```
> 
>  ionic cordova platform add ios
>  ionic run ios
> 
```

Déploiement et tests

ou



Les créations d'entités



Les créations d'entités

Générons une page...

```
> 
>   ionic generate
> 
```

Les créations d'entités

... et routons là !

app-routing.module.ts

```typescript
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

const routes: Routes = [
  {
    path: '',
    redirectTo: 'home',
    pathMatch: 'full'
  },
  {
    path: 'home',
    loadChildren: './home/home.module#HomePageModule'
  },
  {
    path: 'list',
    loadChildren: './list/list.module#ListPageModule'
  },
  {
    path: 'ma-page',
    loadChildren: './ma-page/ma-page.module#MaPagePageModule'
  }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
2 references
export class AppRoutingModule {}
```

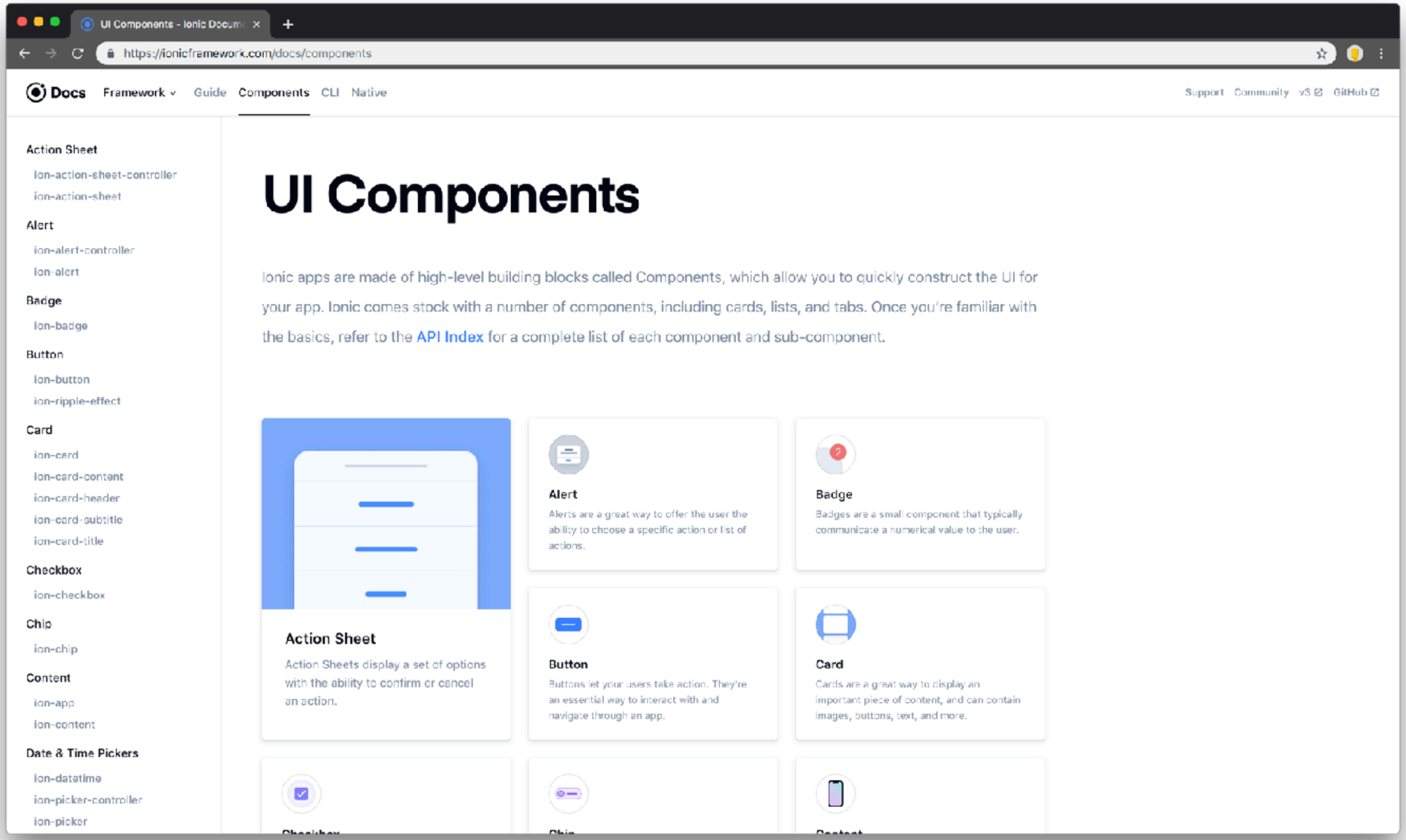
Les créations d'entités

...ajout au menu

app.component.ts

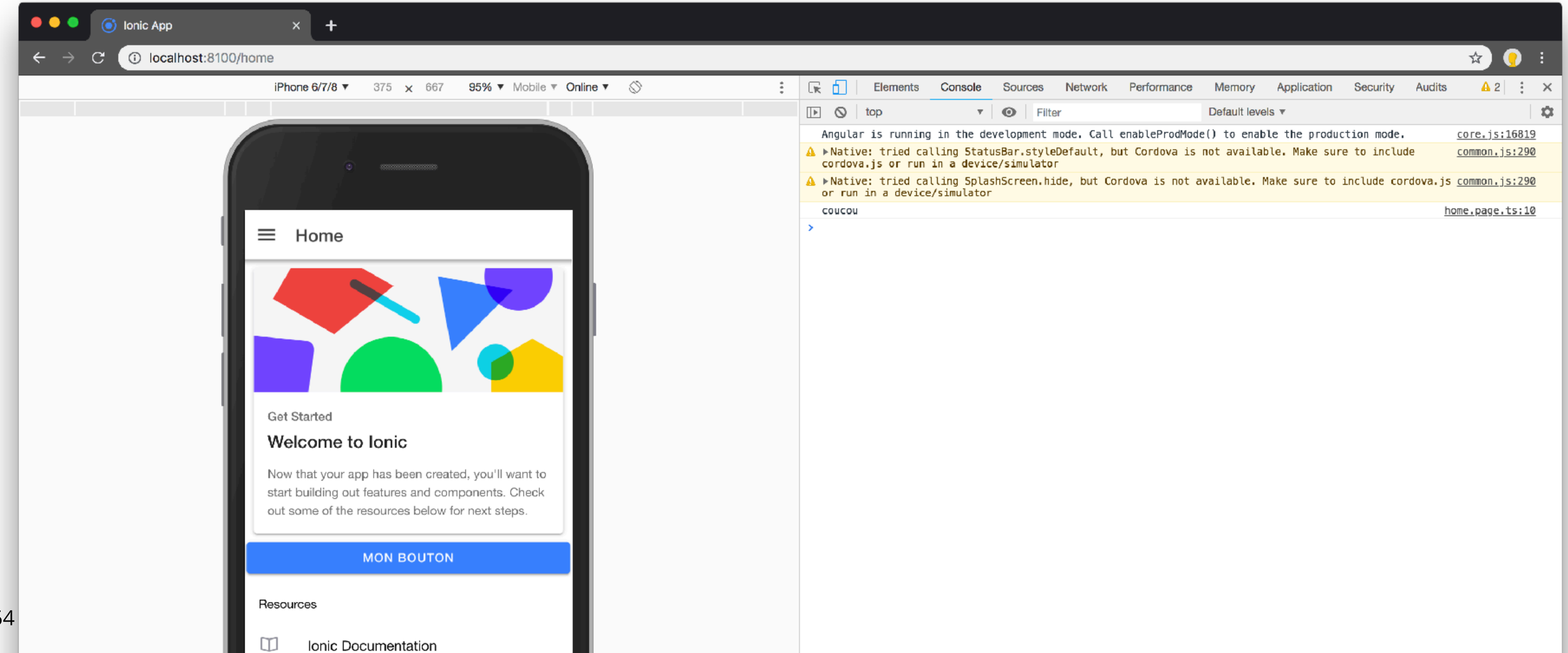
```ts
import { Component } from '@angular/core';

import { Platform } from '@ionic/angular';
import { SplashScreen } from '@ionic-native/splash-screen/ngx';
import { StatusBar } from '@ionic-native/status-bar/ngx';

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html'
})
9 references
export class AppComponent {
  0 references
  public appPages = [
    {
      title: 'Home',
      url: '/home',
      icon: 'home'
    },
    {
      title: 'List',
      url: '/list',
      icon: 'list'
    },
    {
      title: 'Ma Page',
      url: '/ma-page',
      icon: 'list'
    }
  ];

  0 references
  constructor(
    1 reference
    private platform: Platform,
    1 reference
    private splashScreen: SplashScreen,
```

Les components



Les components

Exemple : les ion-buttons



Les components

naviguons

home.page.ts

```ts
import { Component } from '@angular/core';
// import { NavController } from '@ionic/angular';
import { Router } from '@angular/router';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss']
})
8 references
export class HomePage {

  // constructor(private navCtrl: NavController) { }
  0 references | 1 reference
  constructor(private router: Router) { }

  0 references
  public notrePage() {
    // this.router.navigate(['/ma-page']);
    this.router.navigate(['/ma-page', { test: 3456463, back: '/home' }]);
  }

}
```

home.page.html

```html
<ion-header>
  <ion-toolbar>
    <ion-buttons slot="start">
      <ion-menu-button></ion-menu-button>
    </ion-buttons>
    <ion-title>
      Home
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>
    <ion-card class="welcome-card">
      <ion-img src="/assets/shapes.svg"></ion-img>
      <ion-card-header>
        <ion-card-subtitle>Get Started</ion-card-subtitle>
        <ion-card-title>Welcome to Ionic</ion-card-title>
      </ion-card-header>
      <ion-card-content>
        <p>Now that your app has been created, you'll want to start building out features and c
      </ion-card-content>
    </ion-card>

    <ion-button expand="block" (click)="notrePage()">Vers notre page!</ion-button>
</ion-content>
```

Les components

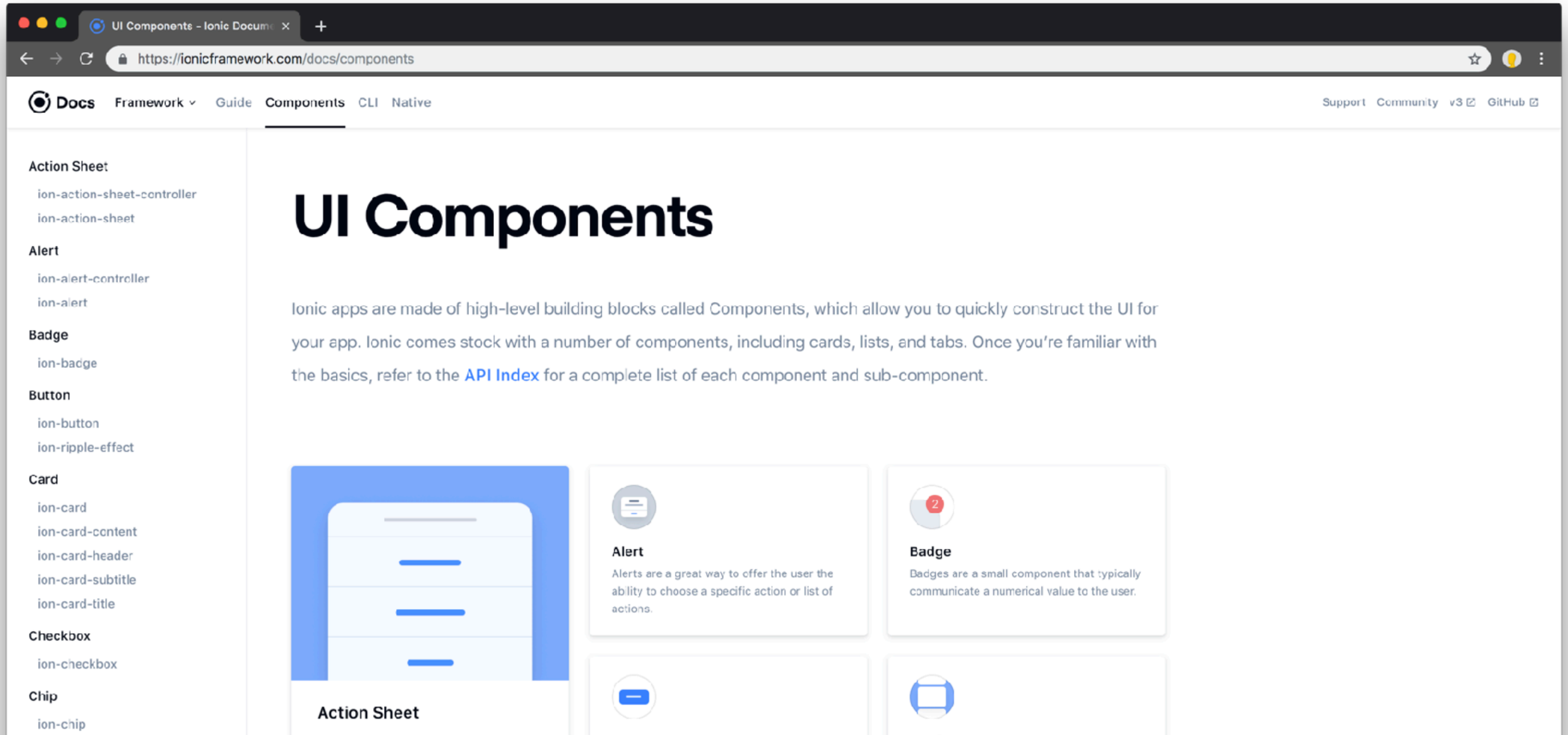
récupération des paramètres

ma-page.page.ts

```
1  import { Component, OnInit } from '@angular/core';
2  import { Router, ActivatedRoute } from '@angular/router';
3
4  @Component({
5    selector: 'app-ma-page',
6    templateUrl: './ma-page.page.html',
7    styleUrls: ['./ma-page.page.scss'],
8  })
   8 references
9  export class MaPagePage implements OnInit {
10
     2 references
11   public prev_page: String = '/list';
12
     0 references | 1 reference | 1 reference
13   constructor(private router: Router, private route: ActivatedRoute) {
14     this.route.params.subscribe(params => {
15       console.log(params);
16       if (params.back) {
17         this.prev_page = params.back;
18       }
19     });
20   }
21
     8 references
22   ngOnInit() {
23   }
24
     0 references
25   public maMethode(event) {
26     this.router.navigate([this.prev_page]);
27   }
28 }
29
```

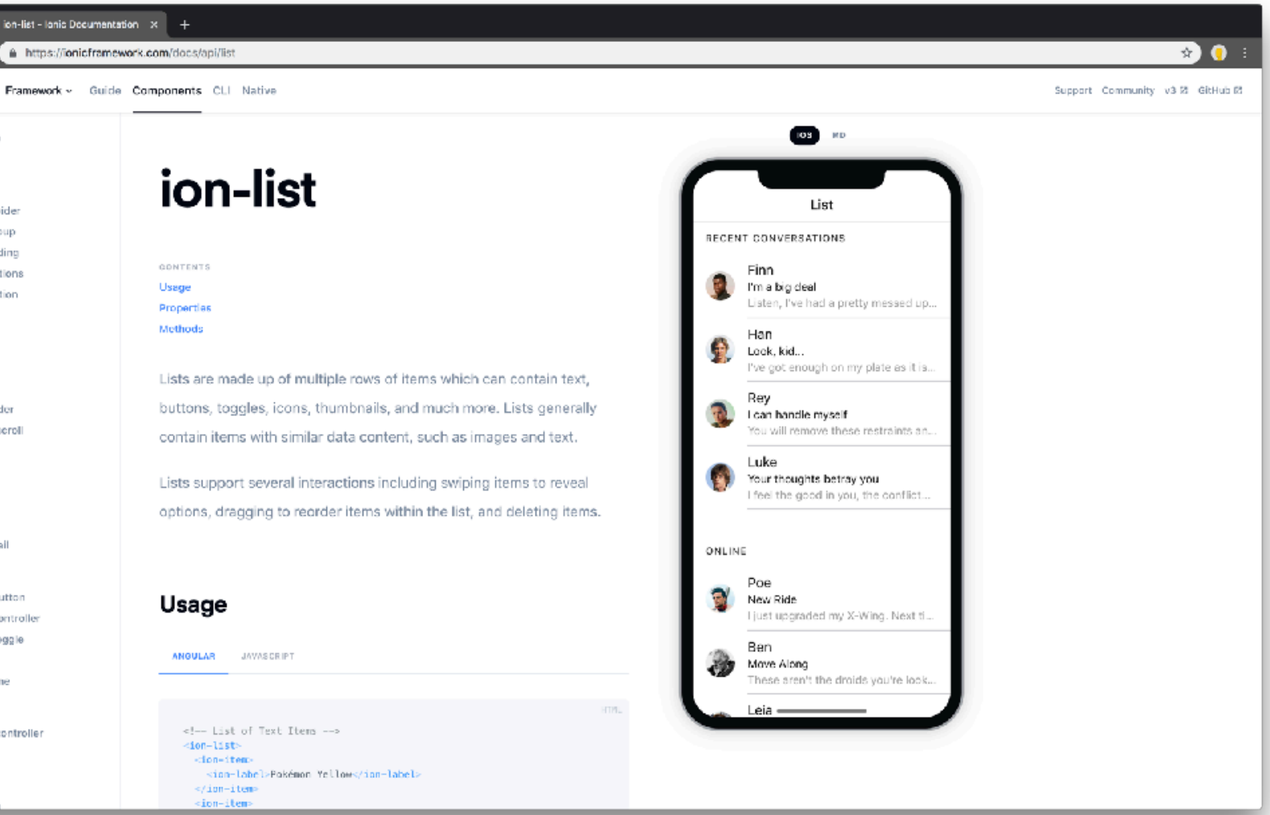
Les components

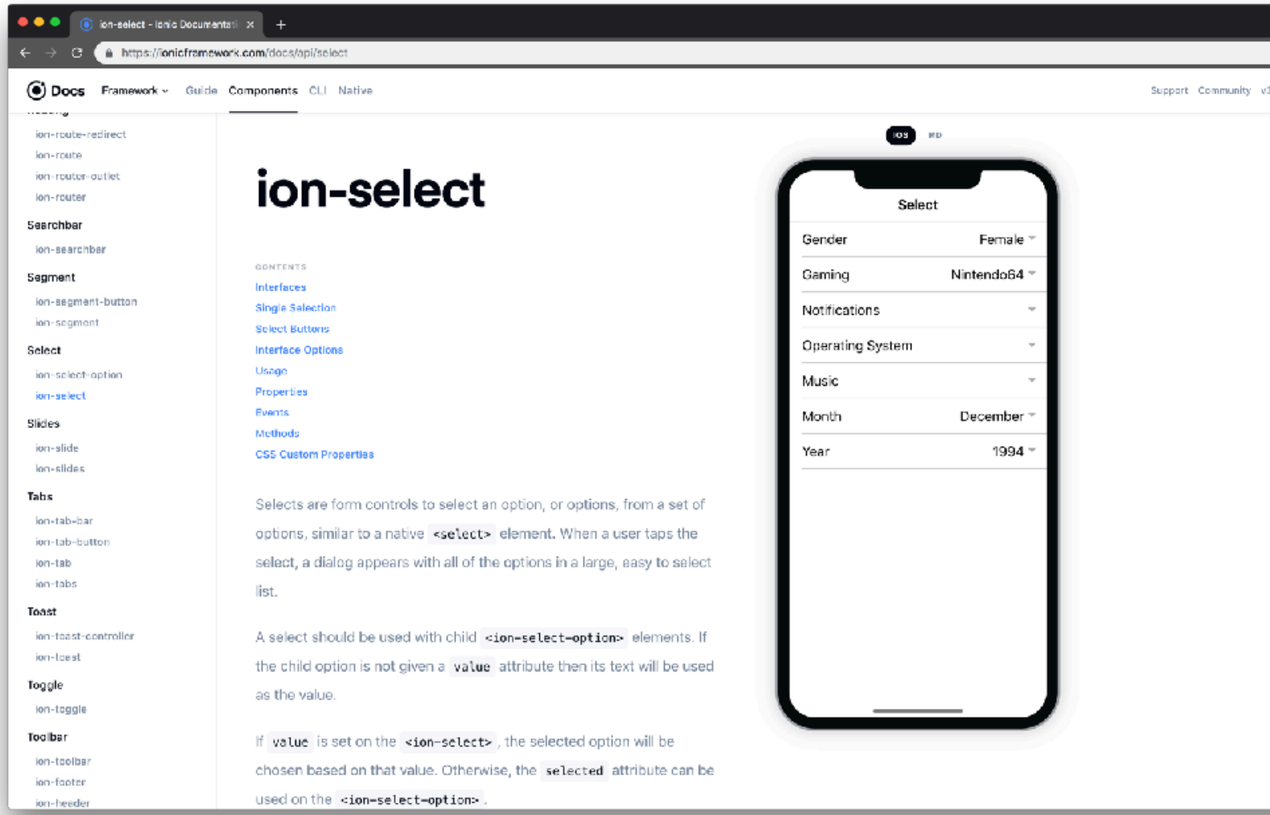
Volontaires ! Création de page, routage, navigation, affichage de composant



Les components

Volontaires ! Création d'un page, routage, navigation, affichage de composant





Exercice